UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES
ARAGON

VNIVERSIDAD NACIONAL AVPNOMA DE MEXICO



# SISTEMA AU[illegible]ADO DE BI[illegible]OTECA DE LA FACULTA[illegible]OS SUPERIOR[illegible] ZARAGOZA


**T E S I S**

QUE PARA OBTENER EL TITULO DE:

**INGENIRO EN COMPUTACION.**

P R E S E N T A N:

**GALVAN VILLALOBOS BALTAZAR.**

**ORTEGA TINOCO RAQUEL MARIBEL.**

ASESOR: ING. AMILCAR A. MONTERROSA ESCOBAR

SEPTIEMBRE 1998.

*Permítanos robarle un cachito de tiempo al tiempo,*
*un pedacito de espacio al mundo,*
*un recuerdo al momento que se va y*
*un sorbo de vida a nuestras vidas...*

*Para agradecer a aquellos*
*que nos dieron cuanto pudieron y más,*
*cuanto pedimos y más,*
*cuanto quisimos y más...*

*A aquellos que nos toleraron,*
*que nos apoyaron y bendijeron...*

*A aquellos que sin nosotros pedirlo nos dieron la*
*oportunidad de vivir y llegar a este punto...*

*Para ellos robamos este instante de vida,*
*para agradecerles.*

*A ellos, nuestros Padres.*

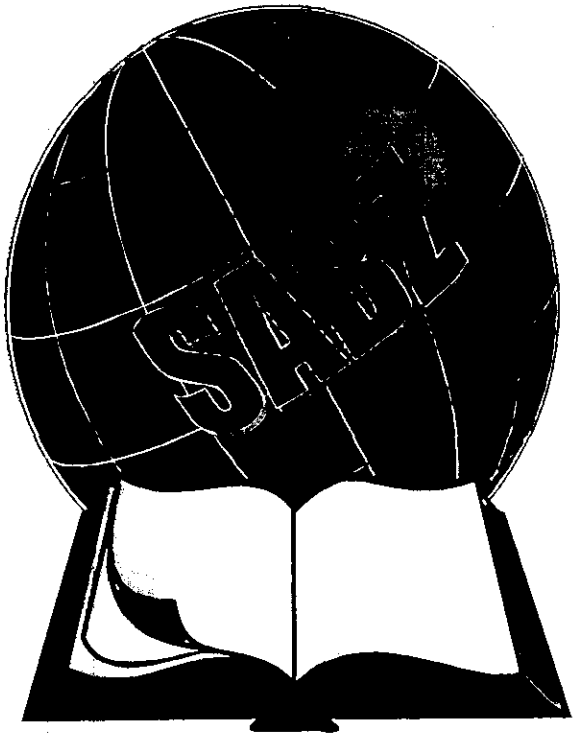
*Si agradecer debemos,*
*debemos agradecer mucho, pero*
*como este libro es un agradecimiento*
*con el, a todos agradecemos.*

*Por último,*
*y por ser el primero*
*Al Creador.*

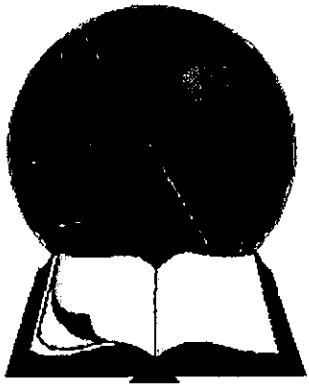
*"Sé que crees que comprendes*
*lo que piensas que he dicho, pero*
*no estoy seguro de que lo que creíste oír*
*sea lo que yo quise decir..."*

**Pressman, Roger S.**

# SISTEMA AUTOMATIZADO DE BIBLIOTECA DE LA FACULTAD DE ESTUDIOS SUPERIORES ZARAGOZA




SEPTIEMBRE 1998.



# TEMARIO

## IV. IMPLEMENTACIÓN

# INTRODUCCIÓN

El Sistema de Automatización de la biblioteca de la Facultad de Estudios Superiores Zaragoza (SABZ), nace a partir de la creciente necesidad de la Universidad Nacional Autónoma de México de realizar paulatinamente un proceso de descentralización, por lo que se ha dado a la tarea de crear una infraestructura tal que permita implantar sistemas computacionales en áreas diversas para la automatización de los procesos delegados a los planteles periféricos.

La Facultad de Estudios Superiores Zaragoza reconoce que las bibliotecas, son un componente fundamental en el desarrollo académico y que requieren por tanto un manejo de información confiable, eficiente y de calidad. SABZ tiene como objetivo primordial ser un sistema capaz de dar los servicios necesarios en las diferentes áreas de consulta dedicadas a los usuarios de las bibliotecas de la facultad (una en campo I y otra en campo II), así como llevar la administración total del resto de los procesos, tales como: procesos técnicos, adquisiciones, inventariado, préstamo, etc., servicios y procesos que actualmente se llevan de manera manual y de forma aislada en cada biblioteca.

Aprovechando el equipo y software de desarrollo con el que cuenta la institución, así como las herramientas visuales y la programación orientada a eventos, el sistema se pretende que trabaje bajo una interface gráfica.

La función esencial en la elaboración de este trabajo, es la de brindar una guía en cuanto al análisis y diseño de un sistema de computo, que pronto se fusionará con uno de mayor envergadura que contendrá información en grandes volúmenes y que la seguridad de su manejo es de gran importancia. Proyectando que en un futuro, SABZ (Sistema Automatizado la Biblioteca Zaragoza) formará parte del macro proyecto de la automatización integral de la FESZ, ya que se interrelacionará con CONEFESZ (Control Escolar de la FES Zaragoza) logrando con esto la integración total en materia de servicios, comunicación y consulta de información

En el capitulo I se presenta la situación a detalle de cuales son las condiciones reales que se viven en la Facultad, así como se exponen los puntos y las bases sobre las que se fundamenta el sistema.

En el Capitulo II se establecen los conceptos y la teoría necesaria para la completa comprensión del análisis y desarrollo del sistema presentado en el capitulo III.

Y finalmente se tiene el proceso de implantación y puesta en marcha en el capitulo IV. Teniendo como primicia que: **"Un sistema será de calidad, cuando cubre eficientemente los objetivos y necesidades que le dieron origen y sobre todo que proporcione la información adecuada a la persona adecuada en el momento adecuado".**

FALTAN PAGINAS

De la: 1

A la: 4

# CAPITULO I

# PLANTEAMIENTO DE LA PROBLEMÁTICA EN LA BIBLIOTECA ZARAGOZA Y PROPUESTA DE SOLUCIÓN.

## I.1 PLANTEAMIENTO DEL PROBLEMA

La Universidad Nacional Autónoma de México como la máxima casa de estudios del país, requiere estar a la vanguardia en cuanto al manejo de información en todos los sectores que la conforman y en realizar una mejora continua en todos los procesos que conlleven a la calidad total de sus servicios.

Para ello la UNAM en los últimos años ha dado un impulso importante a todos y cada uno de sus componentes estructurales, para que en su conjunto conformen la excelencia en la institución.

Uno de estos componentes es la Dirección General de Bibliotecas (DGB), que tiene a su cargo el coordinar el sistema bibliotecario y expedir la normatividad que regirá en todas y cada una de las bibliotecas con que cuenta la institución, esto es, tanto las que se encuentran en Ciudad Universitaria como en todos sus planteles

periféricos (ENEP, Facultades, Preparatorias, CCH), que actualmente suman 164. Esta gran responsabilidad es respaldada por la administración interna llevada a cabo en cada plantel, pero llegan a surgir defazamientos en algunos procesos de información que originan inconsistencia y estancamiento de datos que es solventada hasta el momento en que es realizada la actualización.

Actualmente la DGB como una pieza importante de la UNAM se encuentra en camino hacia la automatización y descentralización de funciones en sus procesos, pero para esto se requiere así mismo de la automatización de las bibliotecas periféricas.

Debido a que las bibliotecas son parte fundamental en todo centro educativo, para la UNAM estas son la columna vertebral que permiten al alumnado poder realizar sus estudios apropiadamente con el apoyo que estas brindan.

Motivados por la situación geográfica del plantel, se tubo la necesidad de subdividir la Facultad en dos secciones denominadas Campo I y Campo II respectivamente, lo que originó indudablemente la necesidad de contar con una biblioteca para cada campo, tomando la primicia de que estas contendrán información especializada de acuerdo a las carreras impartidas en su entorno.

Hoy en día las carreras que se imparten en el campo I son: Técnico en Enfermería, Cirujano Dentista, Lic. en Psicología, Medicina, y la reciente Lic. en Enfermería; por su parte campo II atiende las carreras: Ingeniería Química, Químico Farmacéutico Biólogo y Biología.

Datos reales contabilizan que la biblioteca de Campo I maneja un número aproximado de 83'000 ejemplares, y más de 8'500 usuarios actuales, por su parte

Campo II refiere las siguientes cifras; en cuanto a ejemplares, maneja alrededor de 58'000 y de usuarios reales son 3'098. Los usuarios potenciales de ambos campos son cerca de 10'000.

Las bibliotecas de la FES Zaragoza (campo I y II) actualmente se encuentran en la etapa de transición; mediante el desarrollo de un sistema integral que les permitirá incrementar la eficiencia en sus servicios.

## 1.2 ANTECEDENTES DE LAS BIBLIOTECAS DE LA FESZ

Para lograr una plena identificación de los procesos que se llevan a cabo en las bibliotecas, es invariablemente necesario presentar un análisis a detalle de cuales son las condiciones, requerimientos y en general el ambiente que se respira dentro de la FESZ.

Como plantel periférico, la FESZ ha tenido que sujetarse a lo que determine Ciudad Universitaria; específicamente lo referente a las bibliotecas lo dictamina la DGB y a ajustarse a las disposiciones que esta emita. Hoy en día los procesos administrativos son supervisados por la DGB, así como algunos otros procesos internos que requieren su autorización.

Para tener una visión más completa de la situación actual en la que se encuentran las bibliotecas de las FESZ a nivel informático y de automatización de tareas, se analizarán los aspectos básicos: Software, Hardware e implantación.

- **Software**

Este rubro se refiere específicamente a los programas de computo que actualmente son utilizados para agilizar en cierta medida los servicios propios de la biblioteca.

La circulación bibliográfica, entendiéndose como el movimiento de material bibliotecario, es controlado por el personal de la institución con ayuda de una aplicación denominada CIRCULA que fue elaborada en la DGB con un formato estándar para poder ser aprovechado por todas las bibliotecas de la UNAM que lo requirieran; Sin embargo no abarca todos los procesos requeridos por este plantel, por lo que se complementan los servicios que brinda con procesos manuales.

Por medio de dicha aplicación se llevan a cabo los servicios a los lectores, estos es, registro de usuarios y movimientos del material. Pero sólo en el ámbito de material bibliográfico, es decir libros únicamente, los movimientos de otro tipo de material como lo son audiovisuales, revistas, tesis, mapas, equipo de apoyo a presentaciones, etc., no pueden ser registrados con esta aplicación y tienen que ser administrados manualmente.

La expansión que han tenido en los últimos años los servicios que prestan las bibliotecas y el incremento del material que esta a disposición de los usuarios, hace que el control de estas sea cada vez más complejo, lo que en términos del sistema (CIRCULA) representa una saturación de este, por tal motivo surgen ciertas complicaciones inherentes al manejar la nueva cantidad de información, ocasionando el posible desvió y en el peor de los casos pérdida y/o duplicidad de información.

Los procesos administrativos son manejados independientemente de la pequeña red existente en cada biblioteca, donde el jefe de la misma cuenta con una computadora personal para poder llevar a cabo sus funciones, por lo que sí para la toma de decisiones requiere alguna consulta de información deberá solicitarla al personal y esperar consecuentemente la obtención de la misma, lo cual involucra invariablemente un retraso en procesos críticos.

La consulta por catálogo del acervo de la biblioteca, por parte de los lectores, es otra aplicación disponible que permite una forma ágil, y sencilla de búsqueda de información, el único inconveniente es que la actualización de los datos es controlada por la DGB, por lo que constantemente se llega a suscitar algún atraso de dicha actualización con lo existente realmente en acervo, lo que genera baja confiabilidad en esta fuente de información, provocando que muchos títulos disponibles en anaqueles y que no aparecen en la consulta no sean aprovechados adecuadamente por el lector.

Es importante señalar que el catálogo del material en acervo es independiente para cada biblioteca, es decir, existe un catálogo para el material disponible en la biblioteca del campo I y otro distinto para la del campo II, lo que origina la necesidad de disponer de un equipo de computo dedicado específicamente a la consulta del catálogo de acervo de la biblioteca del campo adyacente.

Los lectores actualmente registrados en una de las dos bibliotecas, deberán si así lo desean, registrarse de igual forma en la otra biblioteca para que puedan beneficiarse de sus servicios (duplicidad de información), debido a que entre ambas bibliotecas no existe comunicación y cada una trabaja independientemente de la otra, pese a que ambas pertenecen al mismo plantel educativo.

Otra aplicación que es manejada actualmente se denomina ADQUNA97, y es actualizada anualmente, donde los últimos dígitos especifican el año de la ultima actualización. Fue elaborada con el propósito de controlar la administración en cuanto a los procesos que involucra la compra de material bibliográfico de procedencia extranjera, pero que de manera similar a CIRCULA fue elaborada de forma tal que fuera aprovechado por todas las bibliotecas periféricas, discriminando las particularidades de cada plantel.

Como se puede apreciar, todos los procesos propios inherentes a su función como soporte estructural en la educación profesional de los estudiantes en las bibliotecas de la FESZ son realizados manualmente y auxiliados con pequeños programas aislados de computo.

Todo lo anterior se traduce en que las soluciones que se habían planteado originalmente ya no satisfacen las necesidades actuales que demandan las bibliotecas y la institución misma, la que indudablemente requiere de disponer de un sistema de control de mayor envergadura, que sea parecido a los sistemas anteriores, en cuanto a la captura y emisión de documentos; adecuándose a los estándares y formatos establecidos y que además tenga la capacidad de contemplar la integración de los procesos actualmente aislados así como el de proporcionar reportes, estadísticas y/o gráficas, dando como resultado mejores tiempo de respuesta y consistencia en la información.

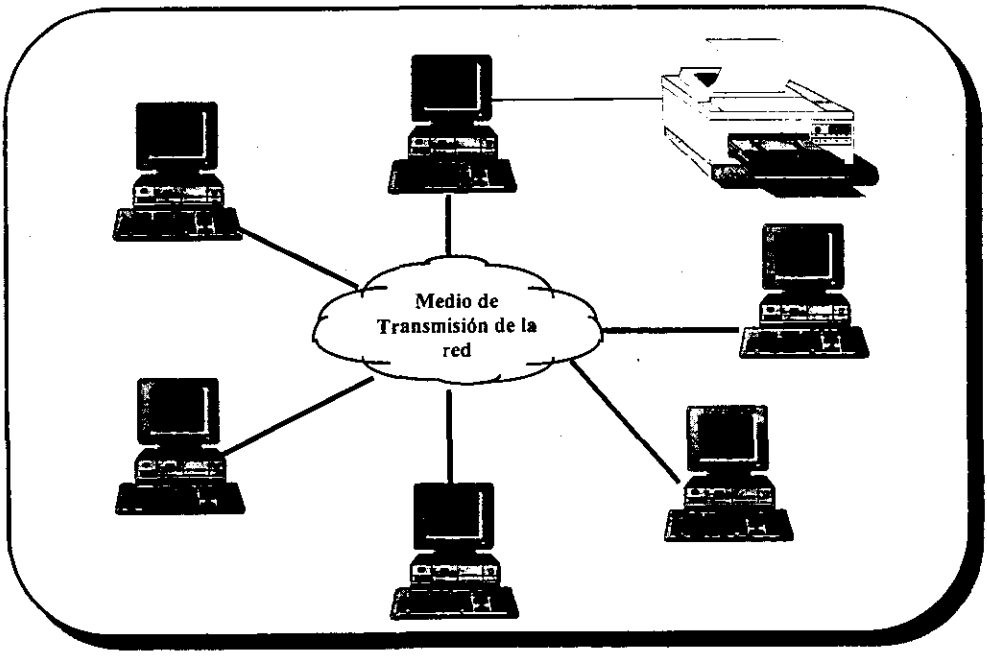
- **Hardware**

Entendiendo como hardware tanto las características técnicas y de infraestructura del equipo de computo existente, así como la arquitectura particular de la red montada en cada biblioteca del plantel.

La FESZ, cuenta con una diversa variedad de equipo disponible para el servicio bibliotecario, ya que hay desde computadoras con procesador 286 hasta equipos Pentium, y que van desde 33 MHZ hasta 130 MHZ; su capacidad de almacenamiento varia desde algunos Kb a Gb, y la memoria RAM oscila entre los 2Mb y los 16 Mb.

En materia de conectividad y desde un punto de vista muy superficial y sin entrar en detalles, una red de computadoras es la interconexión de varias computadoras, donde cada una de ellas interactua con las demás, a través de un medio de transmisión (que puede ser Fibra óptica, Cable coaxial, Par trenzado, Vía telefónica, Microondas, etc.) con el fin de intercambiar información y compartir recursos (archivos, programas, aplicaciones, periféricos, etc.).

**RED DE COMPUTADORAS**



La implantación de una red depende en gran medida del espacio físico en donde se encontrará funcionando, por lo que difícilmente se tendrán redes idénticas en cuanto a disposición de espacios, sin embargo sí compartirán ciertas características tanto técnicas como de aplicación, tal es el caso de las bibliotecas de la FESZ, ambas coinciden con los sistemas de computo que manejan, pero la distribución de su equipo de computo se ajusta a sus requerimientos particulares.

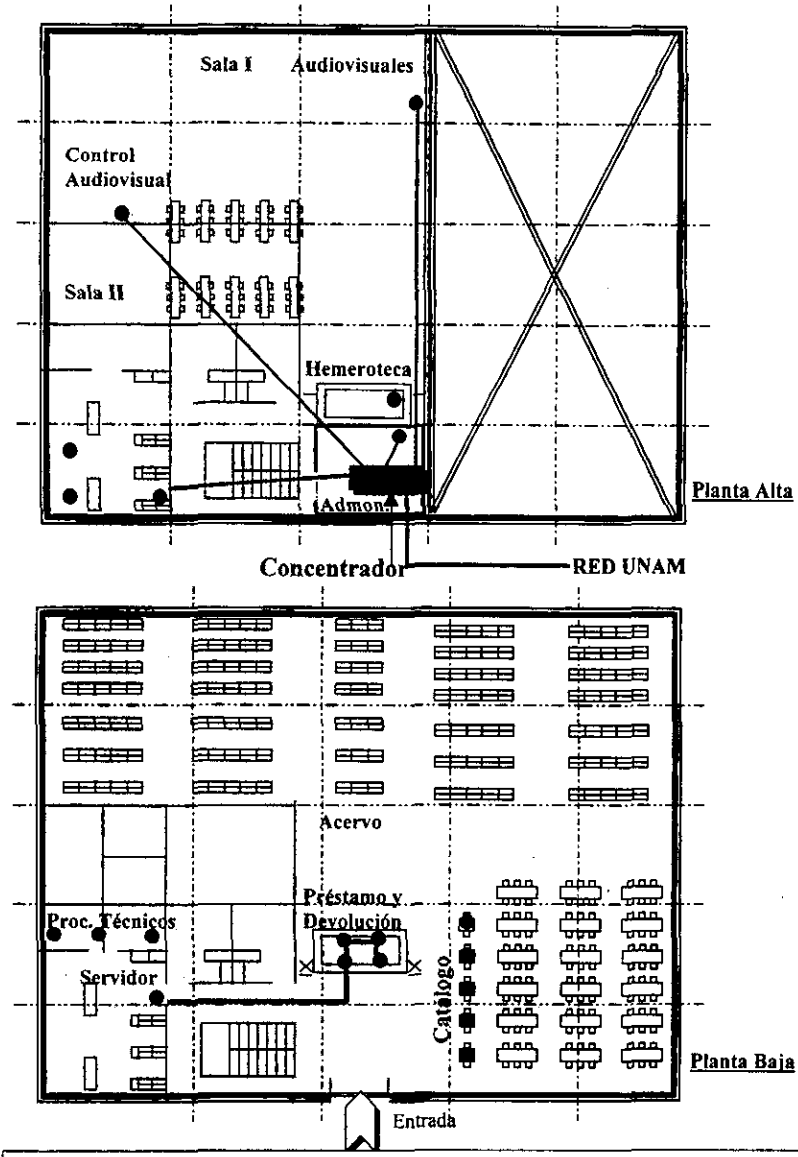
Ahora bien, hay diversas maneras de conectar las computadoras en una red, a esta "***forma de conectar***" computadoras se le conoce como ***Topología física de Red,*** la que básicamente es una descripción de cómo va el medio de comunicación entre un elemento y otro de la red; este puede ser lineal, conectando un punto del edificio a otro distinto, o puede cerrarse sobre sí mismo en forma de anillo o incluso como una estrella, donde los cables salen de un elemento central o concentrador. En realidad un cable lineal puede conectarse de tal forma que parezca cualquier cosa menos lineal, con un anillo puede ocurrir lo mismo, aunque debe cerrase sobre sí mismo en algún momento.

Sin embargo, en la elección de uno u otro tipo de implantación de red influyen algunas características tales como:

- La flexibilidad de la red para expandirse.
- Tráfico máximo de información que puede manejar.
- Tiempos máximos de transmisión y recepción.
- Tolerancia a fallas.

Es evidente que cada topología cuenta con sus características propias y que por medio de un análisis exhaustivo se logra definir cual es la más apropiada en cuanto a funcionalidad y rendimiento para cada situación en particular.

Las siguientes gráficas muestran la distribución del equipo dentro de la biblioteca, así como la descripción de la red existente en cada campo y sus especificaciones técnicas.



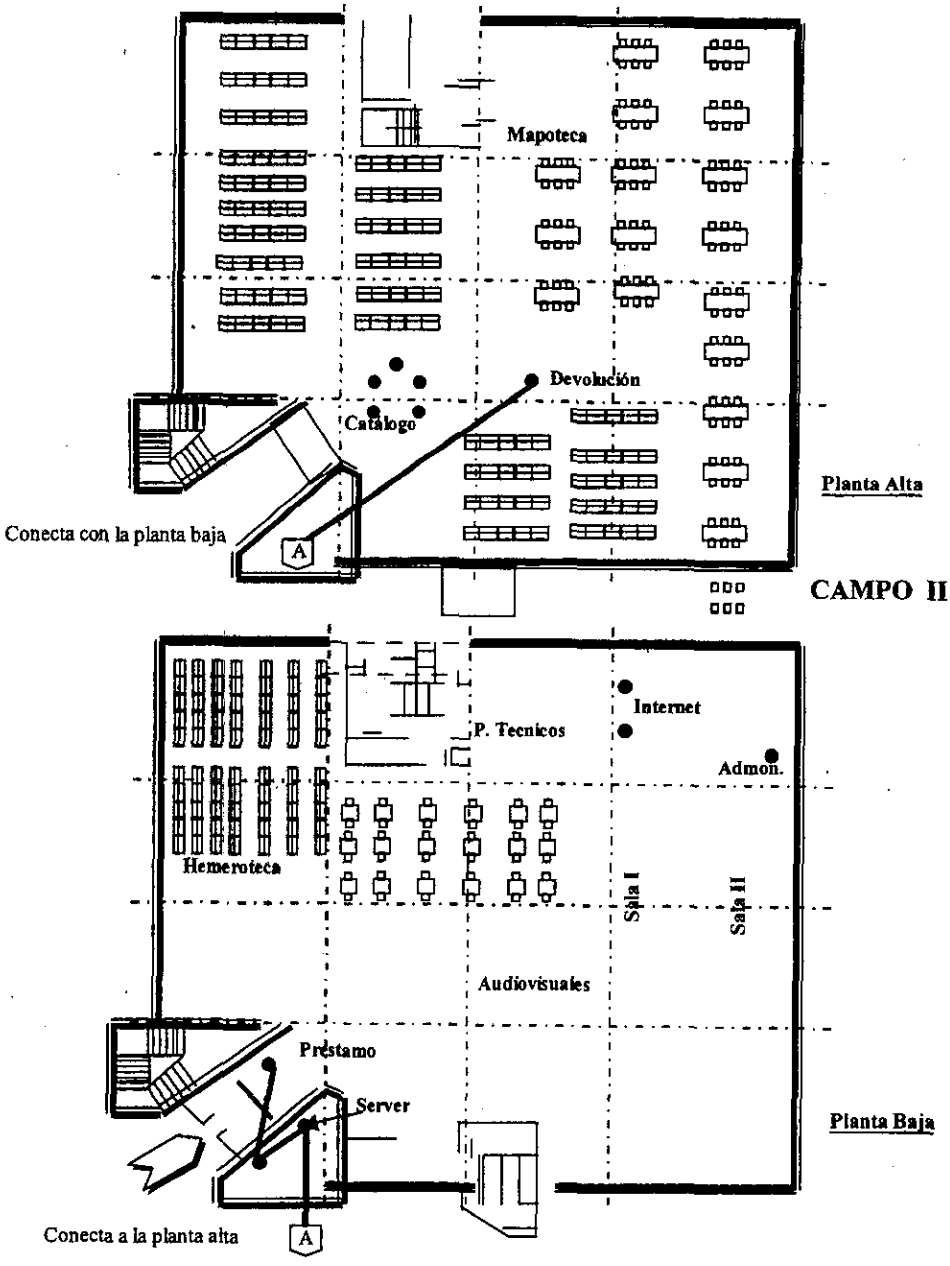

**Facultad de Estudios Superiores Zaragoza Biblioteca Campo I Distribución de nodos de la red Actual**

## Especificaciones del equipo de Campo I:

| Procesador | Memoria RAM en MB | Funcionalidad | Cantidad |
|---|---|---|---|
| 386 | 8 | Servidor | 1 |
| (2)286, (1)386, (1)468 | 4 | Servicios | 4 |
| 386 | 4 | Catálogo | 5 |
| 386,486, Pentium | 4 | Procesos Tec. | 3 |
| 486 | 4 | Hemeroteca | 1 |
| 486 | 4 | Audiovisuales | 1 |
| (1)386, (2)486, (3)Pentium | Variable | Recuperación | 6 |
| 486 | 8 | Administración | 1 |

## Generalidades del Cableado de la biblioteca de Campo I

| | |
|---|---|
| Topología: | Bus |
| Tipo de cable | Thin Ethernet Cable coaxial Delgado RG-58U |
| Tipo de terminación | Conectores BNC |
| Velocidad de transmisión | 10 Mbps |
| Distancia entre terminales | 300m. |
| Impedancia del medio físico | 58.5Ω |
| Normas y estándares | IEEE 802.3a |




**Facultad de Estudios Superiores Zaragoza Biblioteca Campo II**
***Distribución de nodos de la red Actual***

## Especificaciones del equipo de Campo II:

| Procesador | Memoria RAM en MB | Funcionalidad | Cantidad |
|---|---|---|---|
| 586 1GB | 8 | Administración | 1 |
| 486 400MB | 4 | Internet | 1 |
| 586 1GB | 8 | Procesos Técnicos | 1 |
| 386(3)<br>286(1) | 4 | Servicios | 4 |
| 386 | 4 | Catálogo | 5 |
| 486 | 4 | Servidor | 1 |
| 286 | 1 | Apoyo Secretarial | 1 |
| 586 1GB | 8 | Tec. Académico | 1 |
| 8088 | 340Kb | Mapoteca | 1 |

## Generalidades del Cableado de la biblioteca de Campo II

| | |
|---|---|
| Topología: | Bus |
| Tipo de cable | Thin Ethernet Cable coaxial Delgado RG-58U |
| Tipo de terminación | Conectores BNC |
| Velocidad de transmisión | 10 Mbps |
| Distancia entre terminales | 300m. |
| Impedancia del medio físico | 58.5Ω |
| Normas y estándares | IEEE 802.3a |

Las redes anteriores, son redes de tipo bus (lineal) lo que conlleva por sus características técnicas propias de esta topología a presentar redes muy susceptibles a problemas de transmisión, principalmente y debido a que están conectados todos los componentes a un único canal de comunicación (Cable Coaxial[1]), por lo que sí en este existiera una ruptura o falla en el bus de comunicación, la red quedaría dividida en dos ó inutilizada totalmente según el control en la implantación. Si la falla es en una estación de trabajo, sólo repercutirá a esa estación de trabajo en particular. La velocidad máxima que maneja el cable coaxial (que es el utilizado en estas redes) es de 10 Mbps.

**Nota:** *La biblioteca de campo I, cuenta con una raquitica red en estrella (conectada* por medio de un concentrador y cable par trenzado[2]) que es la que permite comunicarse a la RED UNAM, pero que no se involucra en los procesos internos de la biblioteca.

Independientemente de las bibliotecas; la FESZ cuenta con un equipo de computo capaz de sostener sistemas tan complejos y completos que controlen la administración integral de los diversos servicios escolares realizados en la Unidad de Servicio Escolar. Este equipo es una WorkStation SPARC20 con 256 Mb RAM, Disco duro de 4GB, y que trabaja sobre UNIX con Sistema Operativo SunSolaris 2.4. y que interactuará con el sistema automatizado de la biblioteca bajo la primicia de compartir información de alumnos y profesores.

[1] **Cable Coaxial**. Este medio consiste en un conductor central de cobre, rodeado de otro conductor, generalmente una malla de hilos de metal, separados entre sí por medio de un aislante.

[2] **Par Trenzado**. Cable de cobre en dos hilos por los que fluye la información comercialmente existen 5 niveles o categorías, las más usadas son el nivel 3 (10 Mbps) y 5 (100Mbps, voz, datos y video).

Cabe hacer notar que entre ambas redes no existe ningún tipo de comunicación, por lo que trabajan independientemente una de la otra, no obstante que atienden a los mismos usuarios

## 1.3 PROPUESTA DE SOLUCIÓN

Una vez que se han comprobado los grandes beneficios que trae consigo la automatización de procesos, es innegable la necesidad que existe en las bibliotecas de la FESZ de contar con un sistema integral de automatización que involucre todo el conjunto de servicios que brindan, y no menos imprescindible es el de disponer de una comunicación ágil y eficiente a nivel informático entre ambas bibliotecas.

La elaboración consecuente de un estudio de factibilidad, es decir, contemplar las posibles soluciones, elaborando a grandes rasgos un análisis Costo-Beneficio, ayuda en gran medida en la toma de decisiones.

Una alternativa que cubriría de manera instantánea las necesidades de la FESZ, es la adquisición de un sistema diseñado exprofeso para la administración de bibliotecas, que indudablemente requiere de una previa evaluación para cada uno de los sistemas que circulan en el mercado de software.

La FESZ por tanto, considerando esta opción, realizó un estudio de los principales productos que proponían una solución viable a sus requerimientos, de los que se destacan: TinLib, Horizont, en su momento el ALEPH 300 y con mejoras el ALEPH 500.

Algunos de estos sistemas son realmente muy completos, tal es el caso del ALEPH500, sin embargo y dado que pretenden abarcar y funcionar bajo todo tipo de situación, no cubren de manera total las particularidades propias de la institución, además de que su infraestructura es centralizada y la DGB pretende que sea distribuida por lo que los ajustes necesarios al sistema, hoy en día no han sido resueltos. Por otro lado, si la biblioteca en su momento requiere expandirse y cubrir nuevos procesos, así como modificar algunas actividades, deberá solicitar el desarrollo o modificación de algún modulo de manera especifica y particular a la empresa desarrolladora del software. Así mismo, el mantenimiento debe ser realizado por la empresa provocando un costo adicional y una dependencia directa del producto y de la empresa en cuestión.

Otra alternativa a analizar que muy frecuentemente es considerada cuando alguna organización requiere de un sistema, es la creación y en algunos casos la adecuación de un sistema existente que se ajuste a sus requerimientos.

Analizando a detalle las alternativas anteriormente expuestas, esto es, la posibilidad de adquirir un sistema previamente elaborado, o la creación de uno específicamente diseñado para satisfacer las necesidades existentes y tomando en cuenta todo tipo de recursos con que dispone la institución, (financieros, económicos, humanos, tecnológicos, intelectuales, etc.) así como los beneficios directos que se obtendrían por parte de ambas alternativas se concluyó, que la adquisición de un sistema no estaba al alcance de las posibilidades de la institución a nivel de los recursos que demandaba, así como y debido a las características propias inherentes a la situación del plantel, un sistema de bibliotecas estándar no cubre eficientemente las necesidades y perspectivas de sistematización, por lo que un **"Sistema a la Medida"** denominado así porque se origina con el objetivo de cubrir en un 100% los requerimientos de los usuarios finales, es lo que realmente

solventaría y permitiría una completa automatización de estas. El sistema a la medida, nace a partir de una problemática para cubrir ciertas necesidades, crece y se desarrolla bajo estas cláusulas y da como resultado un sistema tal, que abarca los objetivos, requerimientos y perspectivas. Otra ventaja significativa con la creación de un sistema es que permite un desarrollo en paralelo, esto es, por un lado se tiene avance en la elaboración del sistema, y por el otro en la adecuación de la infraestructura de hardware que el sistema demande.

La propuesta de solución que se pretende es la elaboración de un sistema automatizado de bibliotecas de la FESZ, denominado SABZ que sea capaz de proporcionar los servicios necesarios en las diferentes áreas de consulta dedicadas a los usuarios de ambas bibliotecas, y de igual forma permita un control total en la administración del resto de los procesos tales como: ***procesos técnicos, adquisiciones, inventario, préstamo de equipos, catalogación, etc.*** Así como contar con la posibilidad (a mediano plazo) de conjuntarse y formar parte del macro proyecto de automatización integral de la FESZ que consiste en la interrelación con el sistema CONEFESZ (Control Escolar de la FESZ) existente y que permitirá la excelencia total en materia de servicios, comunicación y consulta de información en este plantel.

El sistema SABZ debe ser por tal un software de calidad ya que debe ser confiable, legible y evolutivo, por que si se parte de un software de calidad se obtienen resultados con calidad y para ello debe cumplir con las siguientes características:

- **Conformidad Funcional**.- Es la capacidad de un software de ejecutar las funcionalidades (tareas) definidas en las especificaciones que lo originan.

- **Extensibilidad.-** Es la facilidad de un software para adaptarse a cambios realizados en las especificaciones que en muchas ocasiones se puede considerar reingeniería ya que el análisis del sistema en un momento dado se trabajaría en retroceso.

- **Reusabilidad.-** Es la capacidad de que el software pueda ser utilizable en su integridad o parcialmente en nuevas aplicaciones.

- **Eficacia.-** Es la capacidad de hacer un buen uso de los recursos que se utilizan. Un software eficaz explora al máximo las particularidades del sistema informático sobre el cual es ejecutado, y será muy dependiente de él.

- **Portabilidad.-** La facilidad de que un software pueda ser transportado sobre diferentes plataformas.

- **Verificabilidad.-** Es la capacidad de contar con los procedimientos de validación y aceptar todo tipo de pruebas (test).

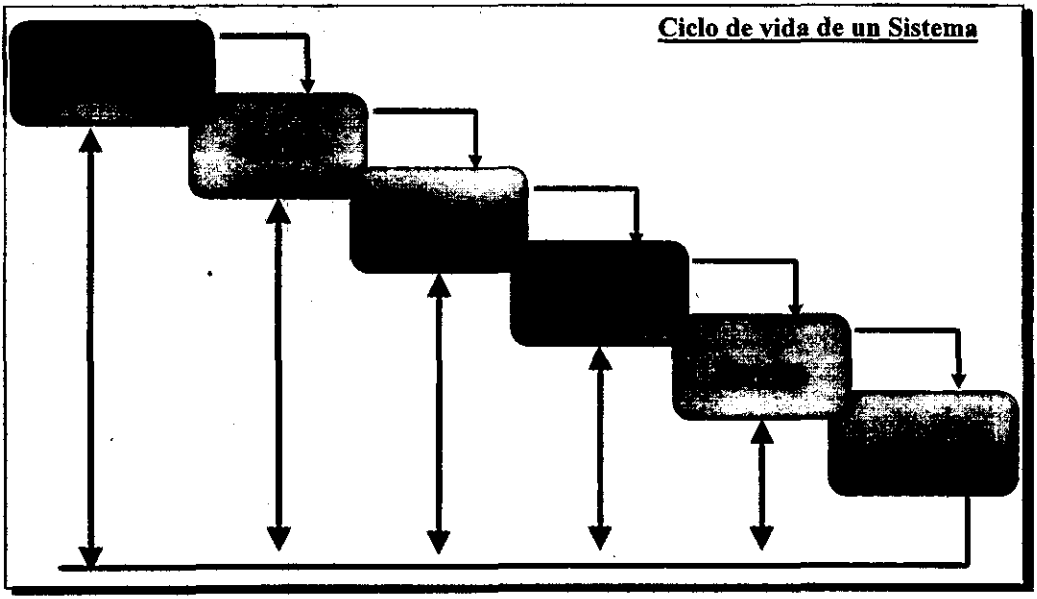
- **Integridad.-** Es la capacidad de un software de proteger sus propios componentes contra los procesos que no tengan derecho a acceder a ellos. Cada entidad debe sistemáticamente prohibir el acceso a la información que ella guarda, a las otra entidades del sistema que no tengan razón de accesar a esta.

- **Facilidad de utilización.-** Donde un software es amigable con el usuario final, que tanto el acceso de datos, la obtención de resultados así como los posibles errores, deben mostrarse de manera legible y clara.

- **Confiabilidad.-** Significa que los usuarios puedan confiar en que el software y el hardware trabajarán según se planeó.

Una vez definida la línea a seguir, en cuanto al desarrollo de un sistema a la medida, se tiene la tarea fundamental de concretar las ideas y expectativas mediante una correcta planeación y estructuración, así como un profundo análisis que contemple todos los elementos que se van a interconectar para lograr los objetivos trazados.

El ciclo de vida de un sistema esta representado por el ***"Modelo en Cascada"***[3] ya que exige un enfoque sistemático y secuencial del desarrollo del software que comienza con el nivel del sistema y progresa a través del análisis, diseño, codificación y mantenimiento.




[3] **Modelo en Cascada**. Técnica que abarca métodos, herramientas y procedimientos para la elaboración de un Sistema de información, facilitando un desarrollo racional y oportuno. (Consideramos que esta técnica es la más apropiada y completa que se ajusta a las proyecciones particulares del Sistema de Automatización de Bibliotecas Zaragoza)

- **Análisis y requisitos de software**. Es el proceso de recopilación de información que conduce a la comprensión de la naturaleza de los programas que hay que construir, así como comprender el ámbito de la información del software, la función, el rendimiento y las interfaces requeridas.

- **Diseño.** El proceso de diseño traduce los requisitos en una representación de software que puede ser establecida de forma que obtenga la calidad requerida antes de que comience la codificación.

- **Codificación**. El diseño debe traducirse en una forma legible (lenguaje entendible) para la máquina.

- **Prueba** La Prueba se centra en la lógica interna del sistema, asegurando que todas las sentencias se han probado y en las funciones externas, realizar pruebas que aseguren que la entrada definida produce los resultados que realmente se quieren.

- **Mantenimiento.** El sistema indudablemente requerirá cambios después de que se entrega al usuario, esto debido a que se hayan encontrado errores, a que el sistema deba adaptarse a cambios del entorno externo o a que el usuario requiera ampliaciones funcionales de rendimiento.

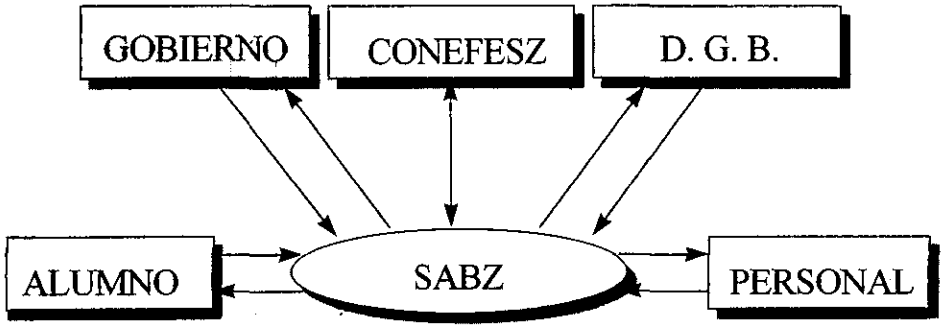
La planeación, por tanto nos permite analizar adecuadamente un problema, diseñar una solución de forma comprensible, elaborar cuidadosamente el código fuente y probar el programa a fondo.

La recopilación de información se realiza mediante la investigación, la entrevista o la observación, para poder determinar objetivos concretos, funcionalidad, procesos, limites, fronteras y sobre todo situarlo en el contexto organizacional de la institución.

Para llevar a cabo este paso, inicialmente se conciertan citas y entrevistas con los usuarios finales y con el personal indicado de las distintas áreas que conforman la biblioteca, así como con los encargados de los departamentos adyacentes que de una u otra forma contribuirán al perfecto funcionamiento de SABZ.

La comprensión total del entorno en donde se desarrollará, permite elaborar y situar los enlaces y flujo de información con los procesos realizados en paralelo y que dependen de información compartida y directamente relacionada con el sistema.

El siguiente diagrama de contexto plasma dichos enlaces.




- **CONEFESZ: Control Escolar de la FESZ.**
- **SABZ: Sistema Automatizado de Biblioteca Zaragoza**
- **D.G.B.: Dirección general de Bibliotecas.**

Establecidos los enlaces del sistema, se comienza con la estructuración, mediante la información que es obtenida previamente, la cual tuvo que ser recopilada, organizada y clasificada, permitiendo una visión global de lo que se pretende abarcar con el sistema.

Es indudable que uno de los puntos más trascendentes y de mayor peso durante el desarrollo de cualquier proyecto son las entrevistas, las que nos permiten obtener información no disponible en documentos y proyectar mejoras en las operaciones actuales, mismas que se verán reflejadas en: una reducción del trabajo de oficina, el mantener una posición competitiva, mejorar la imagen (en cuanto a servicios al cliente y operadores del sistema) y lo más importante, un apoyo substancial en la toma de decisiones.

La identificación de las necesidades es el punto de partida en la evolución de un sistema; para empezar, se definen los objetivos del sistema, la información que se va ha obtener, la que se va ha suministrar, las funciones y el rendimiento requerido.

## Requerimientos de usuarios.

Un requerimiento, es una característica que debe incluirse en un sistema nuevo, pudiendo ser la inclusión de determinada forma de captura o proceso de datos, un tipo de reporte o estadística, etc., por lo que los requerimientos de usuario son generalmente una combinación de lo que se necesita (elementos críticos para la realización) y lo que se quiere (elementos deseables, pero no esenciales), teniendo que realizarse una distinción entre ambos.

## Especificaciones del Sistema.

Describen la función y el rendimiento de un sistema, así como las restricciones que gobiernan su desarrollo. También describe la información (control y datos) que sirve de entrada y salida al sistema, se puede decir, que limitan cada uno de los elementos del sistema asignados.

En esta etapa como en todas las demás, la comunicación intensa con el usuario final es la base para el éxito del proyecto. El usuario (refiriéndose al personal de las bibliotecas, alumnos y departamentos internos de la FESZ) debe comprender los objetivos del sistema y ser capaz de exponerlos claramente, por otra parte, se deben realizar las preguntas adecuadas y hacer una correcta investigación. Dejando las puertas abiertas para futuras modificaciones a lo largo del proceso de la elaboración del sistema, ya sea porque normalmente, es difícil para el usuario establecer explícitamente al principio todos los requisitos o porque quizás surjan nuevos lineamientos que permitan hacer más eficientes los procesos de manera externa (nuevos estándares o modificación de estos) o incluso nuevos caminos que optimizan las propuestas iniciales, a esto se le conoce como ***"Retroalimentación o feedback"***.

Si la retroalimentación se lleva a cabo de manera adecuada se pueden evitar futuras incorformidades y frustraciones tanto para el desarrollador del sistema como para los usuarios, dado que cada cual concibe el sistema de determinada forma y funcionalidad y si en el transcurso de su elaboración no existe comunicación tal vez el sistema llegue a ser totalmente diferente a lo esperado. Es por lo que la retroalimentación es muy importante, puesto que gracias a ella podemos ir adecuando el sistema a las necesidades reales del usuario para que sean más funcionales. No obstante dicha retroalimentación no debe sobrepasar ciertos límites

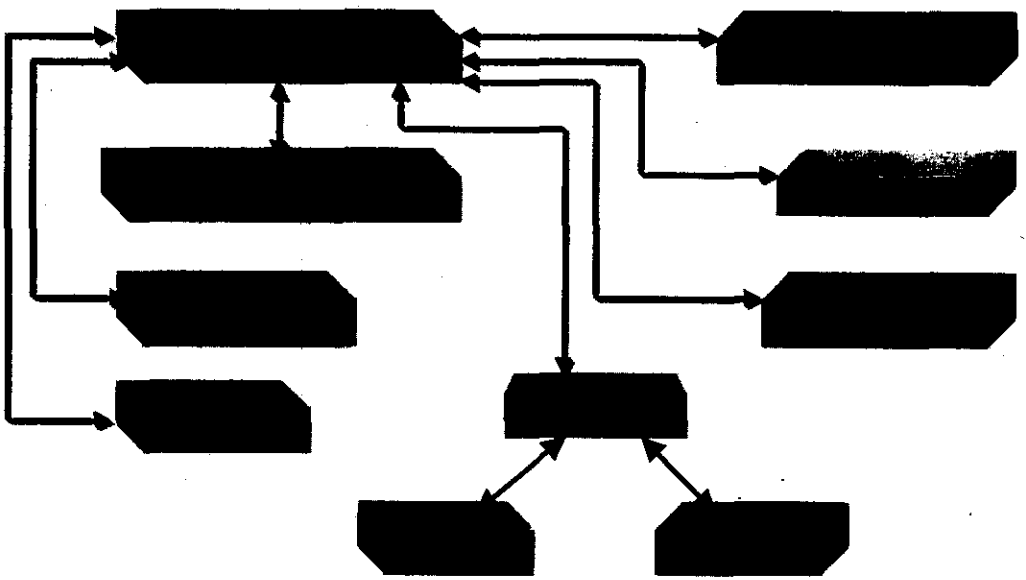
que lleguen a desvirtuar los objetivos iniciales del sistema, pero no tan pobre que no se abarquen estos.

La retroalimentación debe indicar que tan bien está cumpliendo su propósito el sistema de información, en donde los participantes proporcionan declaraciones acerca de la efectividad de este, lo que implica que la retroalimentación debe sistematizarse y convertirse en un aspecto de rutina del control y de las actividades del sistema. El sistema de información debe, por tanto ser capaz de adaptar su desempeño en respuesta a la retroalimentación, diseñado para modificarse con facilidad, en otras palabras debe ser sencillo poder expandirse para acomodar el crecimiento de nuevos tipos de actividades, y fácil de contraerse en respuesta a la eliminación de cierta línea de procesos.

Determinando e identificando el flujo de información que deberá prevalecer internamente en el sistema en cuanto a los módulos que lo conformarán, da la pauta para establecer proyecciones y límites de este, procurando abarcar lo suficiente para satisfacer las necesidades que lo originaron, pero no tan extenso que se pierda de vista la funcionalidad principal y se confunda en procesos secundarios.

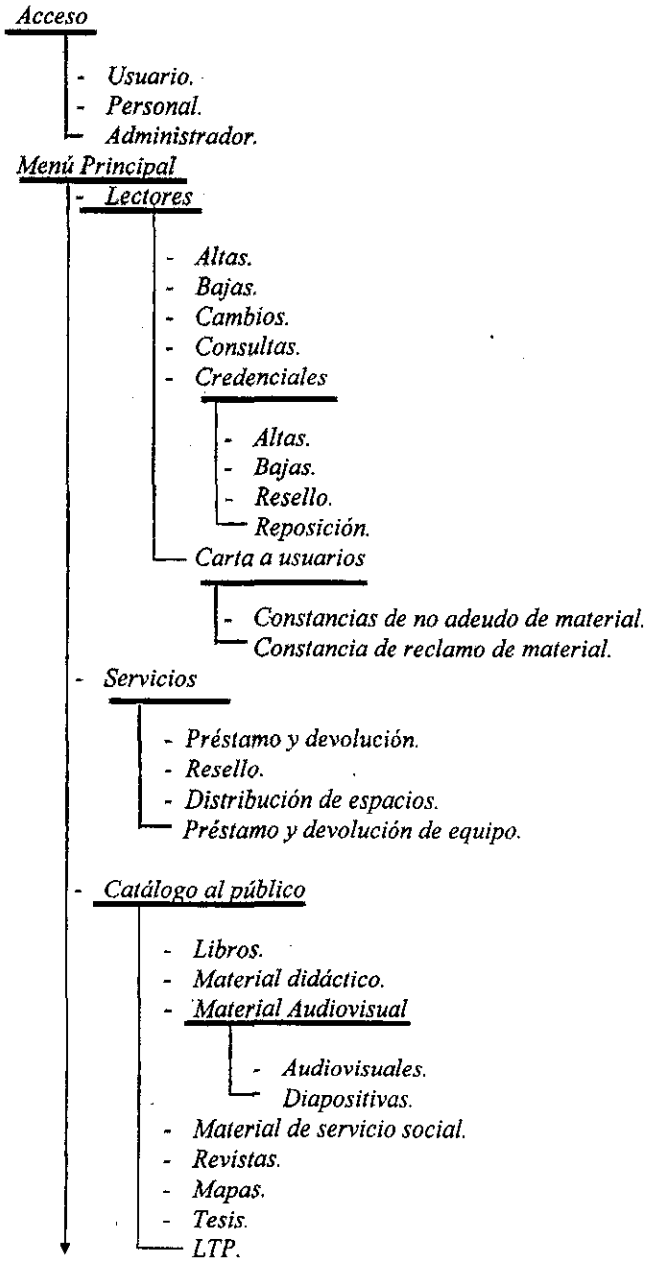
Los módulos que dieron origen los lineamientos anteriores son los siguientes:

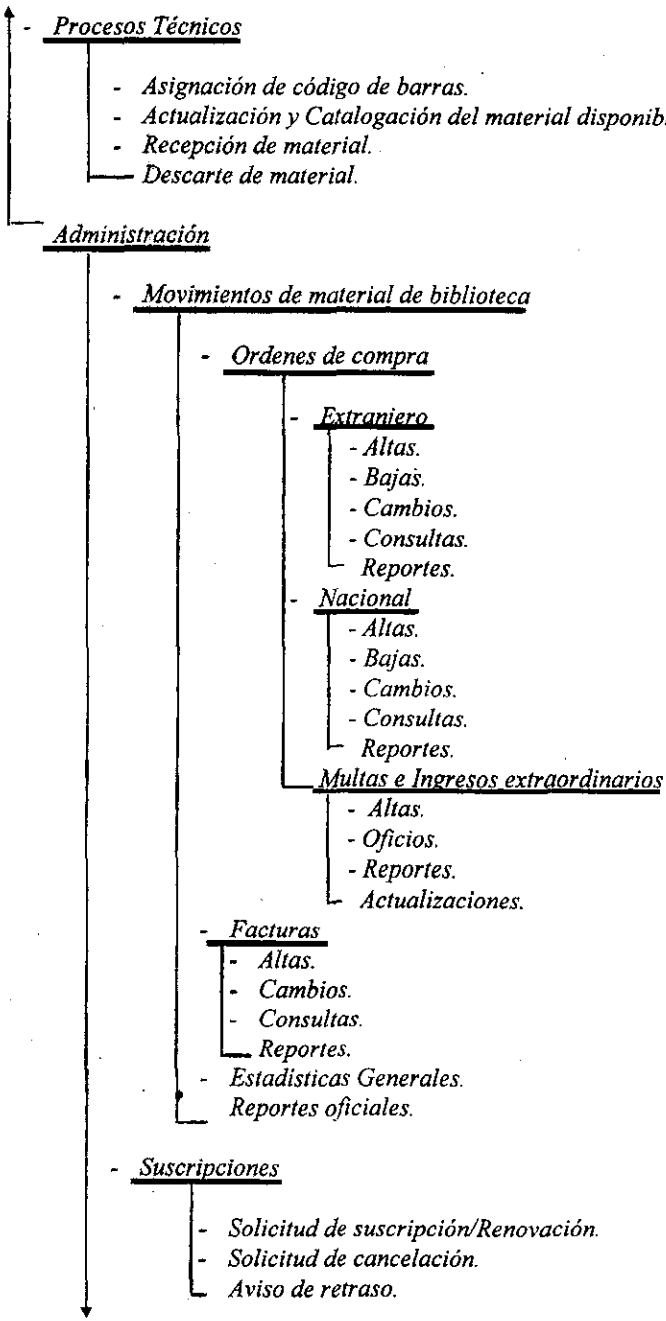
# MODULOS

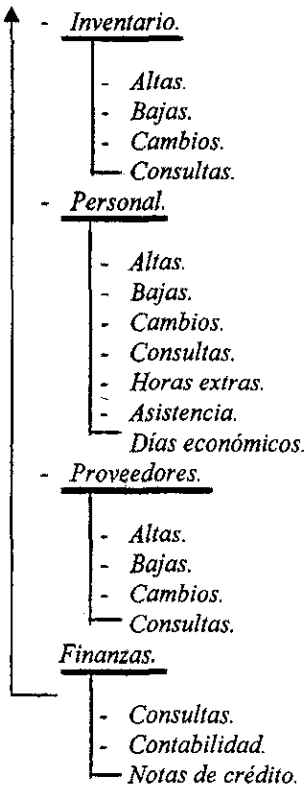


Gracias al cumulo de información obtenida y realizando un análisis detallado, orientado a la optimización de los procesos de la aplicación actual y abarcando los que por el momento se realizan manualmente, se conforma el diagrama de descomposición funcional, este es un mecanismo que permite refinar las tares de procesamiento requeridas por el software para llevar a cabo funciones específicas. Dicho diagrama será la base para el desarrollo del sistema, conteniendo las proyecciones planeadas que abarca SABZ en cuanto a funcionalidad y procesos.

## *Diagrama de Descomposición Funcional*




Procesos Técnicos
Asignación de código de barras.
Actualización y Catalogación del material disponible.
Recepción de material.
Descarte de material.
Administración
Movimientos de material de biblioteca
Ordenes de compra
Extranjero
Altas.
Bajas.
Cambios.
Consultas.
Reportes.
Nacional
Altas.
Bajas.
Cambios.
Consultas.
Reportes.
Multas e Ingresos extraordinarios
Altas.
Oficios.
Reportes.
Actualizaciones.
Facturas
Altas.
Cambios.
Consultas.
Reportes.
Estadisticas Generales.
Reportes oficiales.
Suscripciones
Solicitud de suscripción/Renovación.
Solicitud de cancelación.
Aviso de retraso.




El diagrama abarca los procesos estructurales de SABZ y lo desglosa de tal forma que se logran obtener funciones concretas de estos.

A continuación se presentan los aspectos más importantes del presente diagrama:

***Acceso***. Un aspecto básico y de gran importancia es la seguridad, y por tanto la confiabilidad en la información, la cual se obtiene teniendo un estricto control de los usuarios y niveles de acceso, de tal forma que la aplicación proporcione una interface fácil de usar para los usuarios autorizados que realizan consultas

autorizadas con datos válidos y precisos. Equivale también a una puerta cerrada para usuarios no autorizados y proporciona mensajes informativos y útiles a los usuarios autorizados que hacen consultas no autorizadas, cometen errores o intentan procesar datos no válidos. Esto se realiza por medio de claves de acceso para los usuarios del sistema.

***Lectores***. Corresponde en esencia al servicio directo con el usuario final (lectores que hacen uso del servicio de la biblioteca), abarcando los procesos fundamentales de control (altas, bajas, cambios y consultas) así como los procesos inherentes al manejo y uso de la credencial, identificador del usuario por y para el sistema. Las cartas a usuarios, son aquellos oficios que para algunos tramites administrativos son requeridos por el usuario hacía la institución o viceversa.

***Servicios***. Principio funcional de la biblioteca, consistiendo en el control de los movimientos del material bibliotecario, entendiendo por este tanto libros, revistas, tesis, mapas, LTP (Laboratorio y Taller de Proyectos, realizados por estudiantes de I.Q.), Material de servicios social, Material Didáctico, así como el equipo de audiovisual y salas de proyección.

***Catálogo al Público***. La presentación precisa y actualizada del material disponible para su consulta y uso, incluyendo todas sus modalidades.

***Procesos Técnicos***. Control de los procesos internos y mantenimiento que se realiza para todos y cada uno de los materiales que se accesan a la biblioteca hasta que finalmente puedan estar a disposición de los usuarios.

***Administración***. Comprende los aspectos más delicados y de estricto control, por ser el manejo de dinero y la distribución del presupuesto (partidas)

correspondientes a las bibliotecas, abarca los procesos de ingresos, egresos y del capital disponible. La integran de igual forma la disposición del personal interno de la biblioteca incluyendo también el inventario de la misma.

En concreto, se puede decir que un sistema será de calidad y eficiencia siempre y cuando proporcione la información precisa, en el momento oportuno a la persona indicada.

## I.4 PROPUESTA DE SOFTWARE Y CONECTIVIDAD.

Partiendo de que las necesidades de información de los usuarios determinan en última instancia el tipo de computo, los dispositivos de almacenamiento de datos y el software a utilizar, se debe proponer un software especifico que cubra eficientemente las requisiciones origen.

Hay varios aspectos a considerar al evaluar el software que se utilizará para la elaboración de un sistema. Por una parte el manejador de la base de datos (***Back-end***) que se ocupará del almacenamiento, el resguardo y la estructuración de los datos, y por otra parte, la interface con el usuario (***Front-end*** ) que es básicamente el que se encarga de la plataforma gráfica y cuenta con un lenguaje de programación para realizar los procesos de ejecución, pudiendo ser para ambos el mismo software u otro diferente. Algunos puntos de referencia que ayudan en la toma de decisiones respecto al software son:

1. Calidad del Producto.

2. Evaluación de los beneficios en términos de productividad, derivado del uso de nuevos métodos y herramientas propuestas por el software.

3. Compatibilidad con otros productos y/o con el software de los sistemas existentes que en su caso se relacionarán con el que se elaborará.

4. Que cuente con características tales como:
   - Trabajar en modo multiusuario.
   - Seguridad en el manejo de la información.
   - Niveles de acceso a la Base de Datos.
   - Fácil manejo.
   - Conexión con otras Bases de Datos externas.

5. Manejo eficiente de gran cantidad de registros, dado que el sistema irá siempre en aumento y considerando que muy posiblemente será el manejador para el sistema integral de automatización de la FESZ.

6. Requerimientos de hardware, indispensable para su completo aprovechamiento.

Abarcando, así mismo las funciones, el rendimiento, restricciones, interfaces y la fiabilidad. Las consideraciones de rendimiento incluyen los requisitos de tiempo de respuesta y de procesamiento. Las restricciones por su parte identifican limites del software debido al hardware externo, a la memoria disponible y a otros sistemas existentes.

En requerimientos de procesamiento de datos se refiere al trabajo de detalle del sistema y considera el volumen de datos involucrados, es decir la cantidad de datos que deben procesarse en un periodo dado para obtener cierta información, la complejidad, entendiéndose, como el número de operaciones de datos, intrincadas e interrelacionadas, que se deben realizar para producir alguna información, las restricciones de tiempo, es la cantidad de tiempo permitido o aceptable entre el momento en que los datos están disponibles y el momento en que la información se requiere, y por último las demandas computacionales, las cuales son una combinación única de volumen, complejidad y restricciones de tiempo, para un requerimiento especifico de información.

Considerando los puntos anteriores, y comparando las opciones viables para la institución, se realizó una evaluación completa de los siguientes productos: Delphi 5, Power Builder 5, Progress 8, Visual Basic 4, Sybase, Oracle 8.

Es importante señalar, que tiempo atrás la UNAM tramitó un convenio con la empresa Sybase de México, obteniendo con esto el software pertinente para tener la posibilidad de crear ciertos sistemas computacionales dentro de la institución, la FESZ por tanto, posee este recurso y el impulso que le esta dando a la elaboración de sistemas permite su aprovechamiento integro.

A partir de lo mencionado, y considerando que tan bien satisfacen las opciones deseadas, facilidad de uso, calidad de documentación y sobre todo que este satisfaga los requerimientos en el manejo de la información y basándose en las posibilidades (en todos los aspectos) de la institución, se concluyó que el manejador de bases de datos que cubriría satisfactoriamente y que permitiría una compatibilidad integral con los sistemas existentes desarrollados en C.U. y que tarde o temprano interactuarán con SABZ, es Sybase; en contraparte el encargado de la

interface gráfica de usuario (GUI), será Power Builder 5, que fue desarrollado expresamente para trabajar con Sybase y por tanto la conjunción de estos dos productos resultan ser el más óptimo.

Aprovechando el equipo con el que dispone la institución, así como las herramientas visuales y la programación orientada a eventos, el sistema trabajará bajo un ambiente gráfico lo que representa un sistema amigable, sencillo y claro para el usuario final.

## Conectividad

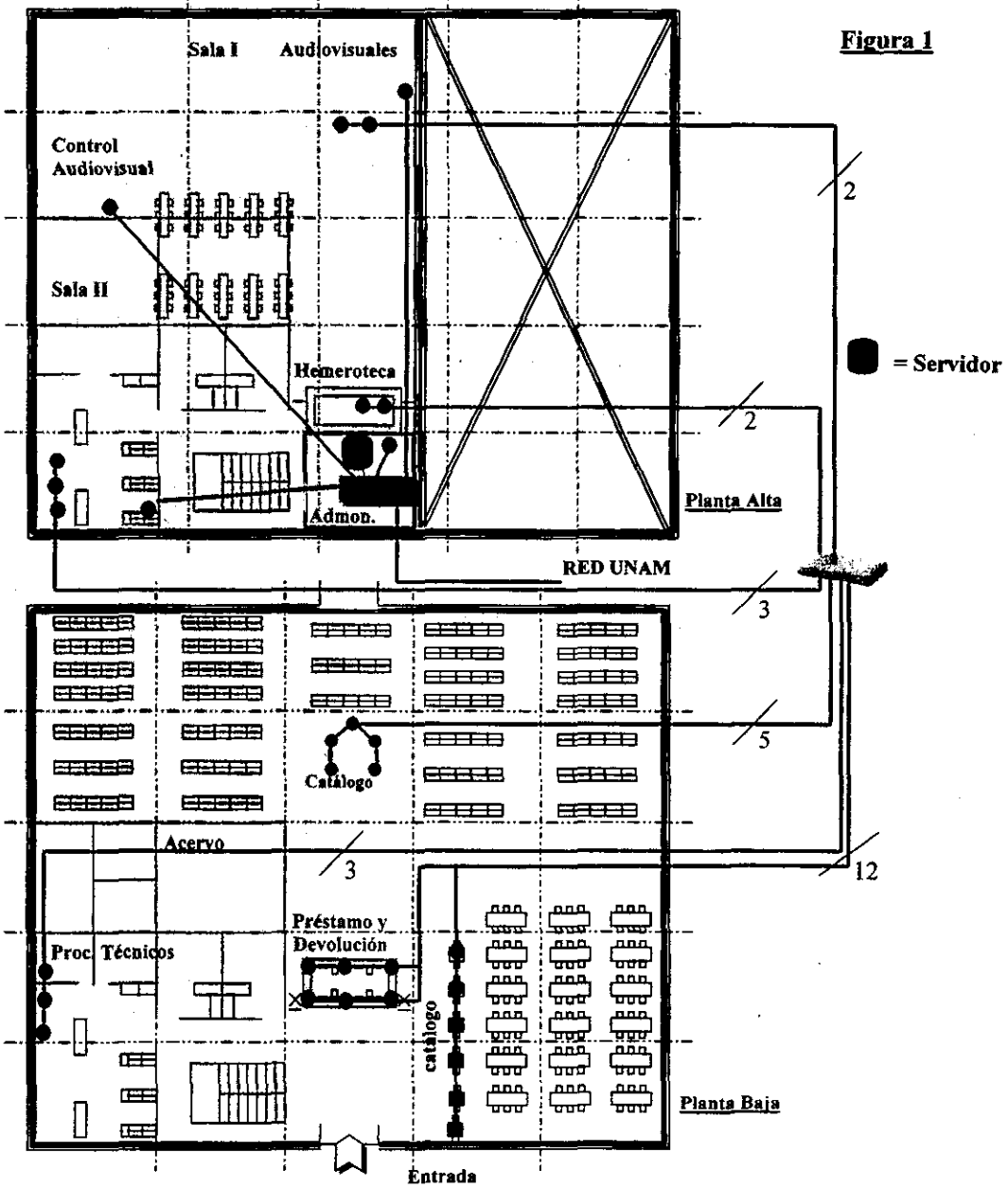
Los recursos clave de un sistema de información y su capacidad de operación están representados por su personal y su tecnología, por lo que la planeación de un sistema involucra de manera determinante el impacto que tendrá este sobre los recursos, asegurándose de que se cuenta con la capacidad suficiente durante el ciclo de planeación no sólo para apoyar las necesidades de operación actuales sino para la operación final y eficiente del sistema en cuestión, así como de futuros proyectos.

Para poder establecer un verdadero servicio de comunicación, control y consulta que permita unificar los recursos y procesos de las bibliotecas, se propone la implantación de los siguientes esquemas de red. (Figura 1 y Figura 2, para cada campo respectivamente).

Debido a las necesidades físicas y características de la red que actualmente se tienen (características citadas anteriormente), se propone la implantación de una red con topología de estrella, donde las conexiones se hacen por medio de un dispositivo central llamado "Concentrador". La red tendrá la característica de transmitir a velocidad de 100 Mbps y se evitará la caída de la red por alguna falla del medio, en

alguna estación de trabajo, o por exceso de colisiones. El medio de comunicación deberá ser cable telefónico par trenzado (UTP nivel 5).

**Figura 1**




**Facultad de Estudios Superiores Zaragoza Biblioteca Campo I**
**Distribución de nodos de la red Propuesta**

**Especificaciones del equipo Propuesto para Campo I:**

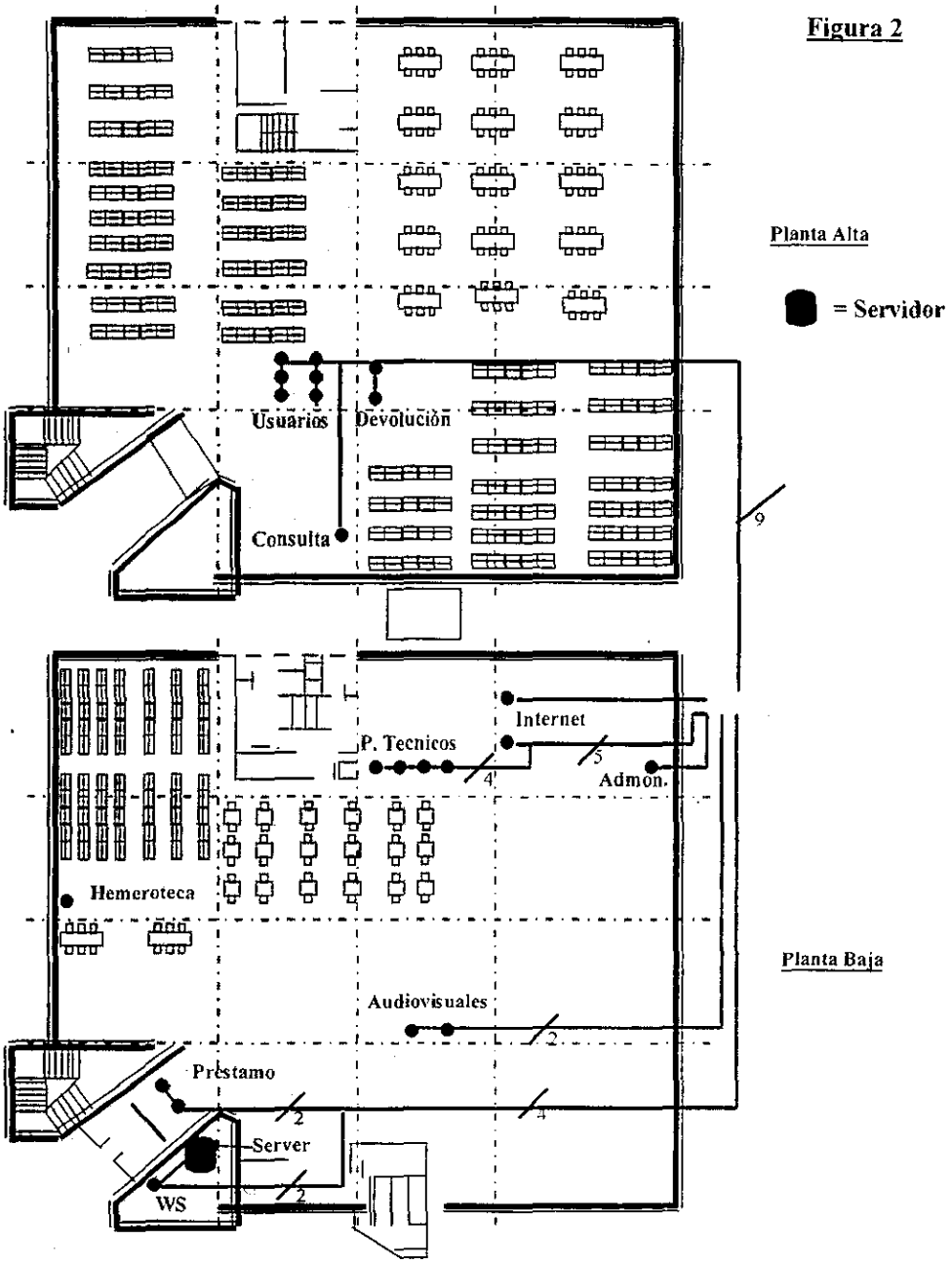
| Procesador | Almacenamiento en Disco duro | Memoria RAM | Funcionalidad | Cantidad |
|---|---|---|---|---|
| Sun Ultra10 17" | 4GB 300MHz o superior | 128 MB | Servidor Primario | 1 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Servicios | 5 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Catálogo | 10 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Procesos Tec. | 3 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Hemeroteca | 2 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Audiovisuales | 1 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Varios | 6 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Administración | 2 |

Con su respectiva interface de red 100baseT para cada estación de trabajo.

**Generalidades del Cableado Propuesto para la biblioteca de Campo I**

| | |
|---|---|
| Topología: | Estrella |
| Tipo de cable | UTP 5 |
| Tipo de terminación | Conectores RJ-45 |
| Velocidad de transmisión | 100Mbps |
| Distancia entre terminales | 100 a 150 metros como máximo |
| Impedancia del medio físico | 100 Ω |
| Normas y estándares | IEEE 802.3a |

**Figura 2**




**Facultad de Estudios Superiores Zaragoza Biblioteca Campo II**
**Distribución de nodos de la red Propuesta**

**Especificaciones del equipo Propuesto de Campo II**

| Procesador | Almacenamiento en Disco duro | Memoria RAM | Funcionalidad | Cantidad |
|---|---|---|---|---|
| Sun Ultra 10 Monitor 17'' | 4GB 300 MHz | 128MB | Servidor de respaldo | 1 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB / 32 MB | Administración | 1 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Internet | 2 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Procesos Técnicos | 4 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Audiovisuales | 2 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Hemeroteca | 1 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Servicios | 3 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Catálogo | 6 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Consulta | 1 |
| Pentium, monitor color. | 1GB 200MHz o superior | 16MB/ 32 MB | Devolución | 2 |

Con su respectiva interface de red 100baseT para cada estación de trabajo.

**Generalidades del Cableado propuesto para la biblioteca de Campo II**

| | |
|---|---|
| Topología: | Estrella |
| Tipo de cable | UTP 5 |
| Tipo de terminación | Conectores RJ-45 |
| Velocidad de transmisión | 100Mbps |
| Distancia entre terminales | 100 a 150 metros como máximo |
| Impedancia del medio físico | 100 Ω |
| Normas y estándares | IEEE 802.3a |

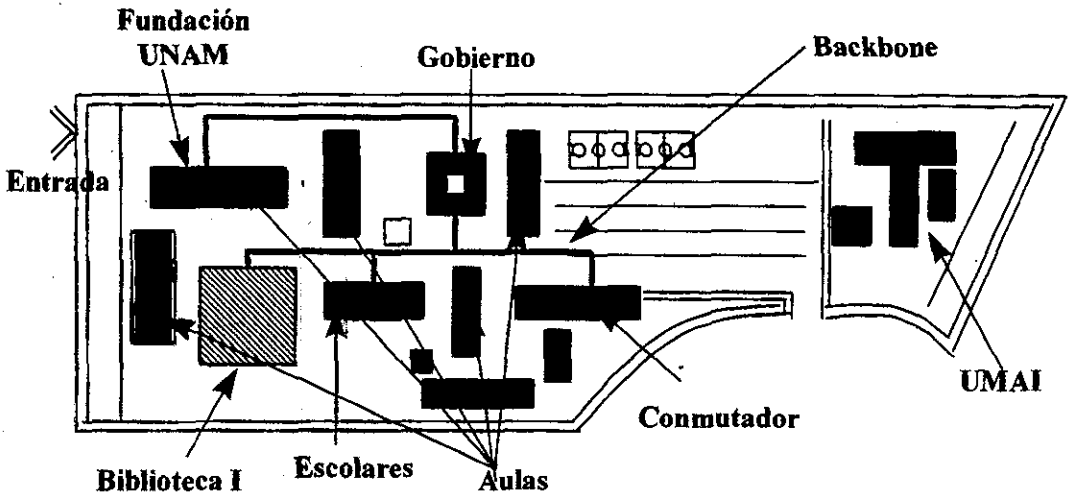
Este tipo de configuración presenta buena flexibilidad para incrementar o decrementar el número de estaciones de trabajo, ya que las modificaciones necesarias no representan ninguna alteración de la estructura central. Así también, la repercusión en el comportamiento global de la red al presentarse una falla en uno de sus nodos o periféricos es muy baja y sólo afectaría el tráfico relacionado con ese nodo y de igual forma si ocurre en el medio de comunicación. Sin embargo es importante considerar su gran inconveniente, el cual es, si se presentara, falla en el nodo central o concentrador, se verían afectadas todas las estaciones de trabajo, (caso poco probable, ya que el concentrador es un dispositivo fabricado con una alta confiabilidad).

Los enlaces centrales (backbone) son conexiones especiales de alto rendimiento que se utilizan para realizar conexiones a larga distancia que permitan la comunicación entre segmentos de red de un conjunto de redes interconectadas. Es recomendable que sea de fibra óptica ya que es el medio más rápido y confiable.

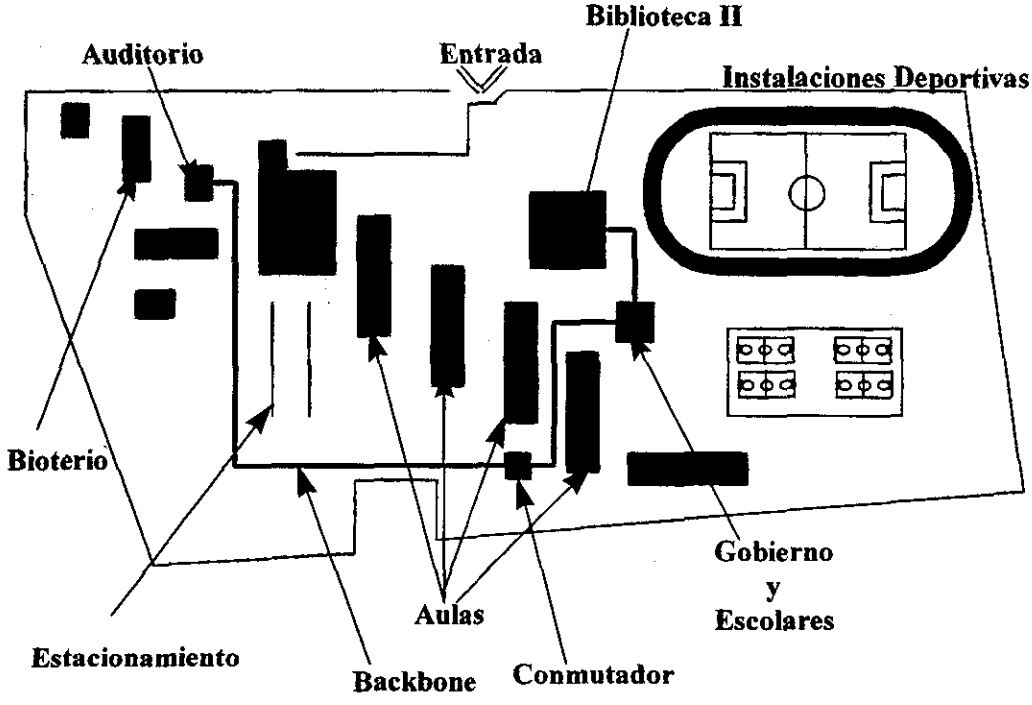
La Fibra óptica, transmite señales de datos mediante luz. La luz modulada pasa por un conductor de vidrio, rodeado por una capa reflejante. Este conjunto esta envuelto en una capa protectora. Las velocidades de transmisión de estas redes se encuentran en el rango de 100 Mbps (ya que depende de muchos factores), pero en aplicaciones especiales se ha llegado a tener velocidades del orden de Gbps. El cable de fibra óptica no resulta afectado por interferencias, resultando muy confiable en casos de alta seguridad.

Básicamente se tiene una distribución y conexión de la red en cuanto al backbone en cada campo, mostrada en la siguiente gráfica:

## CAMPO I




## CAMPO II



Como se mencionó inicialmente, se tiene la urgente necesidad de conectar ambos campos, y dado que el sistema dará servicios a los dos, es indispensable que exista la transmisión de datos entre ellos, para conformar la integridad en la información, esta conexión se realizará mediante TELMEX S.A. de C.V. que proporcionará el servicio de conexión a través de su cableado telefónico mediante una conexión RDI (Red Digital Integrada). De igual forma la interconexión con C.U., y a su vez con Internet se realizará a través del ruteador[4] ubicado en Campo I. Sin embargo y dada la gran cantidad de información que se genera en ambos campos hacía C.U. se sugiere que se disponga de otro canal semejante para evitar que se sature el medio de transmisión y la conexión se vuelva extremadamente lenta. El enlace se conformará de la manera especificada en la **Figura 3.**

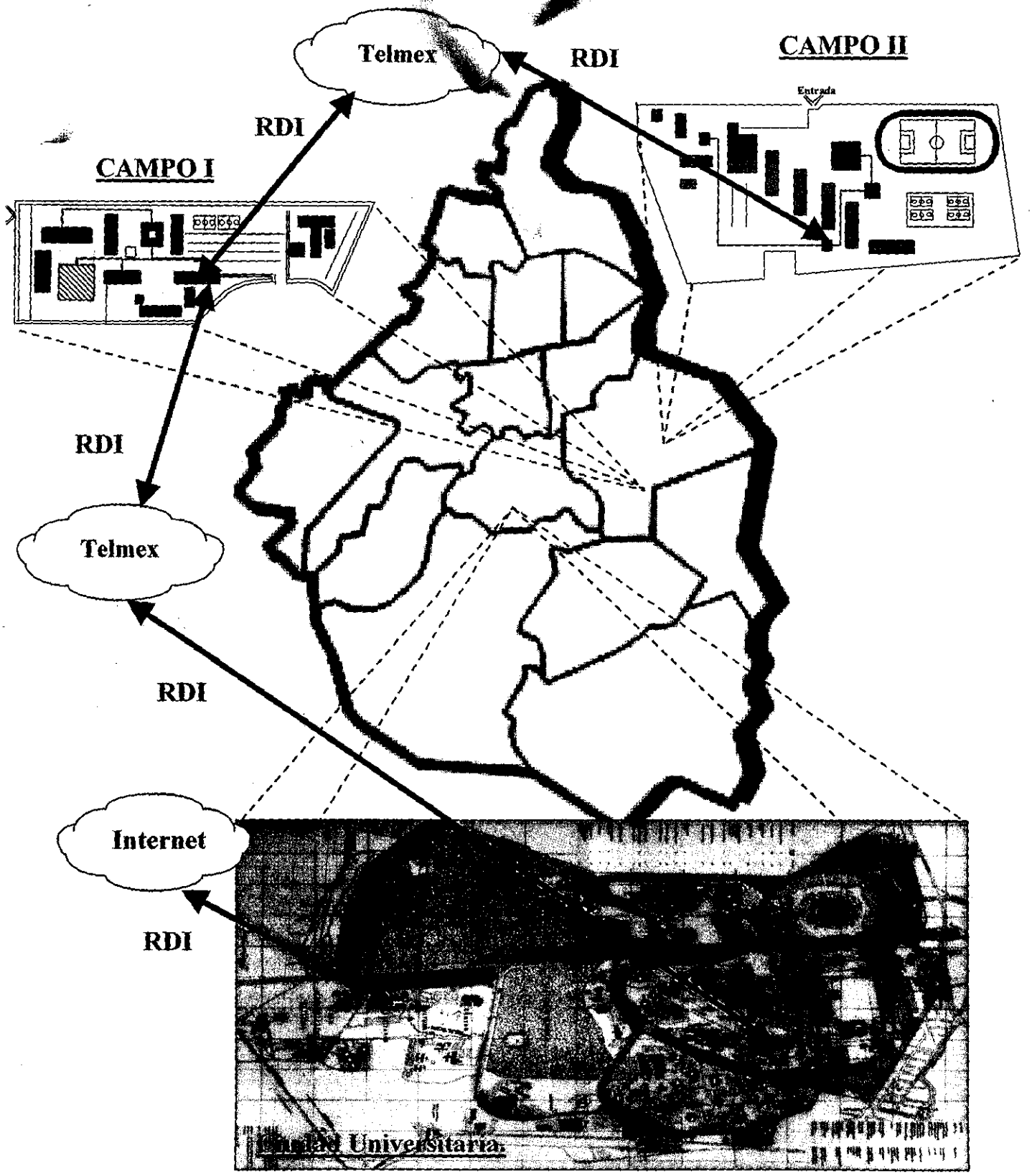
Cabe señalar que durante el proceso total de la elaboración de un sistema es muy recomendable contar con una agenda de trabajo, la que permitirá organizar y administrar de manera eficaz los recursos, los procesos y el tiempo. Esto es, se pueden estructurar los procesos que se puedan realizar en paralelo, organizar los que requieran ser secuenciales, estimar tiempo de cada etapa y considerar si es posible tiempos de holgura aprovechándolos en procesos que presentan mayor dificultad que la esperada; resultando una estimación en tiempo real de la realización del sistema.

Admitiendo que frecuentemente la calendarización de la agenda no se sigue al pie de la letra la mayoría de las veces, sin embargo aun así se convierte en un recurso de evaluación de avances, pudiendo trabajar en su momento en aquellos aspectos que requieran de mayor atención.

[4] **Ruteador.** Es un sistema utilizado para transferir datos entre redes que utilizan un mismo protocolo (conjunto de normas y reglas que permiten una comunicación). Un ruteador o router puede ser un dispositivo software, hardware o una combinación de ambos.

Finalmente, se puede concluir que el desarrollo de un sistema es un proceso que consta del análisis de un sistema, el diseño de uno nuevo o las modificaciones a uno ya existente, la adquisición del hardware y software necesarios y el hacer que el sistema nuevo o modificado trabaje.

# CONEXIÓN PARCIAL DE LA UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO



# CAPITULO II

# FUNDAMENTACIÓN TEÓRICA

Antes de pasar a la etapa de diseño, es importante tener presentes ciertos conceptos básicos, que son aquellos que dan origen a la elaboración y comprensión del sistema, y que otorgan los cimientos que estructurarán el software de aplicación.

## II.1 INGENIERÍA DE SOFTWARE

El término Ingeniería de Software se refiere al conjunto de disciplinas utilizadas para especificar, diseñar y programar software de computación. En otras palabras, son aquellas disciplinas interrelacionadas que son necesarias para construir una empresa computarizada basada en sistemas de datos.

El punto de partida o básico de la Ingeniería de software corresponde a los datos a almacenar y mantener por las computadoras así como la información que emana de dichos datos.

La información está compuesta de datos, imágenes, texto, documentos y voz, a menudo entrelazados de forma muy compleja, pero siempre organizados en un

contexto significativo y útil, que es comunicada a un receptor que la utiliza para la toma de decisiones.

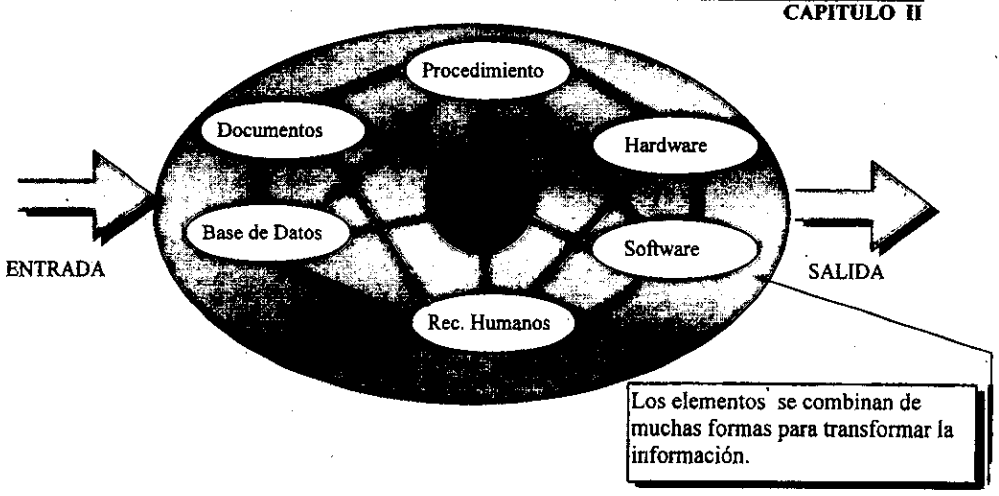
El objetivo básico de la Ingeniería de software es la lógica empleada en los procesos automatizados.

## SISTEMAS DE INFORMACIÓN.

La palabra "sistema" es posiblemente el término más sobre utilizado y del que más se ha abusado en el léxico técnico. Por definición, un sistema es una colección de elementos y procedimientos que interactúan entre sí para lograr un objetivo, pero en cuanto a un sistema de información se debe entender que es: "***Un conjunto u ordenación de elementos organizados para llevar a cabo algún método, procedimiento o control mediante el procesamiento de información.***"

Sin embargo, la información que presenta un sistema debe ser información de calidad, la cual descansa sólidamente sobre tres pilares: Exactitud, Oportunidad y Relevancia, atributos claves de la información. La exactitud significa que la información este libre de errores, clara y que refleje adecuadamente el sentido de los datos en los que se basa. Información oportuna es el hacer llegar la información a los receptores dentro de un marco de tiempo necesario. Y finalmente, la relevancia, conlleva a que se tenga la información adecuada para cada receptor, conscientes de que la información relevante para cierta persona, no lo es necesariamente para otra.

Sin importar las organizaciones a las que sirven o la forma en que se desarrollan y diseñan, todos los sistemas de información están compuestos por los siguientes elementos:




**Elementos de un sistema.**

| ELEMENTO | DESCRIPCIÓN |
|---|---|
| Procedimientos. | Los pasos que definen el uso específico de cada elemento del sistema o el contexto procedimental en que reside el sistema. |
| Documentos. | Manuales, impresiones, y todo tipo de información descriptiva que explica el uso y/o operación del sistema. |
| Hardware. | Dispositivos electrónicos que proporcionan la capacidad de computo y los dispositivos electromecánicos que proporcionan las funciones del mundo exterior. |
| Bases de Datos. | Colección grande y organizada de información a la que se accede mediante el software y que es una parte integral del funcionamiento del sistema. |
| Software. | Los programas de computadora, las estructuras de datos y la documentación asociada, que sirven para realizar el método lógico, procesamiento o control requerido. |
| Recursos. Humanos | Personal capacitado, que realizará el análisis, desarrollo e implantación del sistema. |

A medida que fluye la información en un sistema, esta se transforma. El sistema acepta entradas en una gran variedad de formas, aplica los elementos de hardware, software y humanos para transformar la entrada en salida, produciéndola en varias formas.

La entrada representa a todos los datos que entran al sistema de información, los métodos y los medios por los cuales se capturan e introducen. La entrada está compuesta de transacciones, solicitudes, consultas, instrucciones y mensajes. La transformación puede ser desde una sencilla comparación lógica, hasta un complejo algoritmo numérico o un mecanismo de reglas de inferencia de un sistema experto. La salida puede ser el encendido de un diodo de emisión de luz (LED) o un informe de 200 páginas. Efectivamente podemos crear un modelo de flujo para cualquier sistema de computadora, independientemente del tamaño y de la complejidad.

Una característica compleja, pero muy útil de los sistemas de información es que los elementos que lo conforman, así como el sistema mismo en su totalidad pueden también representar un elemento de un sistema mayor, a estos elementos se les denomina "***macroelementos***" (caso que se aplica a SABZ, que a su vez se planea sea un macroelemento del sistema de automatización integral de la FESZ).

La base de datos es el componente estructural clave en el diseño de sistemas de información. Es la primera fuerza de información de una organización, pero ¿qué se entiende como base de datos?

# II.2 BASES DE DATOS RELACIONALES

Las bases de datos se dice que son una colección de datos interrelacionados y almacenados en conjunto sin redundancias perjudiciales o innecesarias; su finalidad es la de servir a una aplicación o más, de la mejor manera posible, los datos se almacenan de tal forma que queden independientes de los programas que los utilicen, y que emplea métodos bien determinados para insertar, borrar o modificar los datos.

Básicamente una base de datos se define como una colección integrada de datos; los datos contenidos se escriben y se almacenan en algún tipo de formato estructurado. Una base de datos relacional es un conjunto de tablas relacionales que contienen datos, se dice que es relacional, ya que relaciona datos de una tabla con datos de otra. Permite combinar datos de dos tablas para dar respuesta a una misma cuestión. Existen nombres alternativos para describir los elementos que constituyen a las bases de datos relacionales:

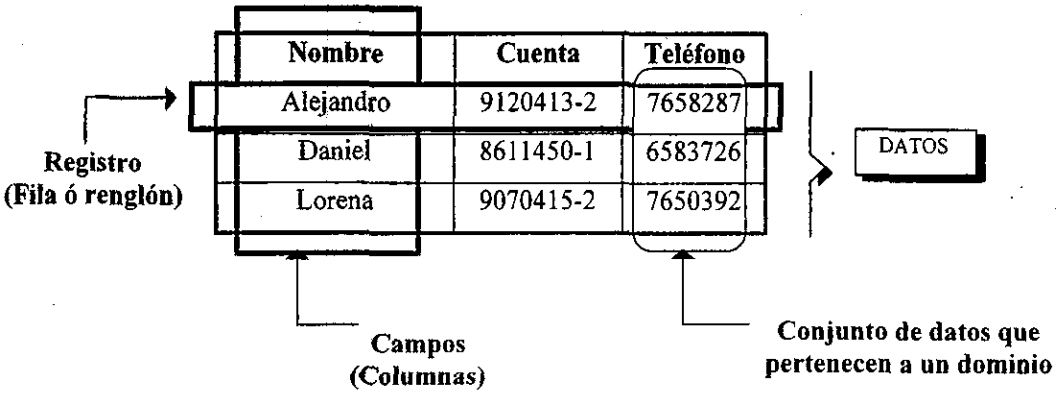
| Terminología Relacional | Representación Física | | SQL |
|---|---|---|---|
| **Afinidad** | **Archivo** | ⇒ | **Tabla** |
| **Tupla** | **Registro** | ⇒ | **Fila (Renglón)** |
| **Atributo** | **Campo** | ⇒ | **Columna** |

Una tabla consta de dos dimensiones (Filas y Columnas), cada fila (renglón) en la tabla tiene datos que pertenecen a alguna cosa o parte de alguna cosa, cada columna contiene datos referentes a un atributo. Sin embargo, una tabla debe cumplir con ciertas restricciones, tales como: las celdas deben ser de valor único

(no grupos o arreglos), los ingresos en las columnas deben ser del mismo tipo, cada columna posee un nombre único en la tabla y no es importante el orden. No pueden ser idénticas dos filas y en estas, tampoco es importante el orden.

Las columnas dan nombre al tipo de datos que estarán contenidos en la tabla. Básicamente pertenecerán a un dominio, el cual es un conjunto formado por todos los posibles valores atómicos de elementos de un dato.

TABLA : **LECTORES**




Se puede decir, por tanto que una tabla será relacional si y sólo sí cumple con las siguientes propiedades:

1. Cada intersección de columna-renglón deberá tener un solo valor atómico.
2. Los valores de los datos en las columnas son del mismo tipo, es decir, deben pertenecer al mismo dominio, a un conjunto de valores que la columna puede tener.
3. Cada renglón es único. No pueden existir dos renglones que contengan valores idénticos.
4. Cada columna deberá tener un nombre único

5. La secuencia de las columnas (izquierda a derecha) no es significativo.
6. La secuencia de los renglones (arriba abajo) no es significativo.

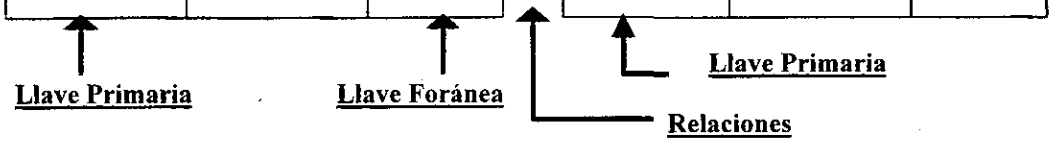
Como quedó establecido anteriormente, cada fila de la tabla es única, esto es, debido a que cada fila contiene una y sólo una forma de identificación para permitir su selección, llamada clave o ***llave***. Si una columna no es suficiente como identificador único, se deberán combinar varias columnas para formar la llave, denominada por tanto **llave compuesta**, de entre todas las llaves de una tabla se elige una que se va a utilizar como identificador principal de la tabla y será conocida como **llave primaria**, la cual debe cumplir con la condición de que ninguna de sus columnas pueda tomar valor nulo. Las llaves restantes de la tabla que no se designaron como primarias, suelen llamarse **llaves secundarias**.

**TABLA : LECTORES**

| Cuenta | Nombre | Tipolec |
|---|---|---|
| 89056523 | Carlos | 2 |
| 90121345 | Elena | 1 |
| 90124536 | Salvador | 3 |

**TABLA: TIPOSLEC**

| Tipolec | Descripción | Limlib |
|---|---|---|
| 1 | Alumno | 4 |
| 2 | Académico | 6 |
| 3 | Administrativo | 6 |




Las llaves permiten el enlace de información de diferentes tablas, esto es, la llave primaria de una tabla puede localizar datos en otra tabla. Una **llave foránea** o ajena es una llave primaria de una tabla que se utiliza para acceder a los datos de una tabla diferente. **"Las asociaciones entre llave primaria y llave secundaria son el nexo de unión que mantienen las bases de datos relacionales"**.

# NAVEGANDO ENTRE TABLAS USANDO LLAVES PRIMARIAS

**AUTHOR**

| au_id (PK) | au_lname | au_fname | address | city | state |
|---|---|---|---|---|---|
| 172-32-1176 | White | Johnson | 10932 Bigge Rd. | Menlo Park | CA |
| 238-95-7766 | Carson | Cheryl | 589 Darwin Ln. | Berkeley | CA |
| 267-41-2394 | O'Leary | Michael | 22 Cleveland Av. #14 | San Jose | CA |
| 274-80-9391 | Straight | Dean | 5420 College Av. | Oakland | CA |
| 341-22-1782 | Smith | Meander | 10 Mississippi Dr. | Lawrence | KS |
| 409-56-7008 | Bennet | Abraham | 6223 Bateman St. | Berkeley | CA |
| 427-17-2319 | Dull | Ann | 3410 Blonde St. | Palo Alto | CA |
| 472-27-2349 | Gringlesby | Burt | PO Box 792 | Covelo | CA |
| 486-29-1786 | Locksley | Charlene | 18 Broadway Av. | San Francisco | CA |

**AUTHOR_TITLE**

| au_id (PK) | title_id (PK) |
|---|---|
| 172-32-1176 | PS3333 |
| 238-95-7766 | PC1035 |
| 267-41-2394 | BU1111 |
| 267-41-2394 | TC7777 |
| 274-80-9391 | BU7832 |
| 409-56-7008 | BU1032 |
| 427-17-2319 | PC8888 |
| 472-27-2349 | TC7777 |

**TITLE**

| title_id (PK) | title | type | price | pub_id (FK) |
|---|---|---|---|---|
| BU2075 | You Can Combat Computer Stress! | business | 2.99 | 736 |
| BU7832 | Straight Talk About Computers | business | 19.99 | 1389 |
| MC2222 | Silicon Valley Gastronomic Treats | mod_cook | 19.99 | 877 |
| MC3021 | The Gourmet Microwave | mod_cook | 2.99 | 877 |
| MC3026 | The Psychology of Computer Cooking | UNDECIDED | | 877 |
| PC1035 | But Is It User Friendly? | popular_comp | 22.95 | 1389 |
| PC8888 | Secrets of Silicon Valley | popular_comp | 20 | 1389 |
| PC9999 | Net Etiquette | popular_comp | | 1389 |
| PS2091 | Is Anger the Enemy? | psychology | 10.95 | 736 |

**PUBLISHER**

| pub_id (PK) | pub_name | city |
|---|---|---|
| 736 | New Moon Books | Boston |
| 877 | Binnet & Hardley | Washington |
| 1622 | Five Lakes Publishing | Chicago |
| 1756 | Ramona Publishers | Dallas |
| 9901 | GGG&G | München |
| 9952 | Scootney Books | New York |
| 9999 | Lucerne Publishing | Paris |

Existen dos reglas que ayudan a asegurar la integridad de los datos: ***la integridad de la entidad y la integridad referencial***, donde la primera de estas establece que una llave primaria no puede tener valores nulos, la segunda llamada ***regla de integridad referencial*** considera la asociación entre las llaves primarias y las llaves foráneas, ya que establece que una llave foránea debe existir en una tabla que contenga la llave primaria, esto significa, que cuando se elimine algún registro y por consiguiente un dato identificador como llave primaria, se deberán eliminar consecuentemente todos los datos que hagan referencia a el por medio de llaves foráneas en otras tablas. Suponiendo que tenemos la tabla **Lectores** que tiene como llave primaria el campo cuenta, y se elimina algún lector, se tendrán que modificar a continuación todas aquellas tablas que hagan referencia a aquel lector por medio de su cuenta, ya que si no, se pierde la integridad referencial dado que existirán elementos de otras tablas que hacen referencia a un lector que ya no existe.

Una vez que los datos quedan estructurados en la base de datos, debe ser posible encontrarlos y leerlos (Proceso de recuperación), sin embargo y por definición, las filas de una relación no están en ningún orden en particular, por lo que para encontrar datos específicos en una base de datos se pueden leer las filas de manera secuencial o empleando un índice. Se puede utilizar un índice para acceder más rápidamente a las filas de una tabla. Un índice es un archivo que contiene una lista de los números de filas organizados por el valor de una o más columnas en cada fila y en ocasiones por los datos o un extracto de los datos. Un índice puede también ordenar los datos contenidos en una tabla.

Para formar y decidir que atributos constituirán los índices se deben considerar cierto enfoque: Evitar los índices que resultarían extremadamente largos (eligiendo atributos que contenga un pequeño conjunto de valores posibles), pero desafortunadamente los atributos que pertenecen a este conjunto no son muy selectivos. Sin embargo, para resolver esto, se deben combinar atributos para la

indización, precisamente como ellos serían combinados para fijar el criterio de búsqueda. (Los índices se basan a menudo en la combinación de dos o más atributos).

La ventaja de los índices es que aumentan la velocidad de acceso a los datos en una tabla, su desventaja es que la adición de datos es lenta, ya que el índice debe ser actualizado con los datos añadidos. Las bases de datos también llegan a tener múltiples índices de manera que las filas se encuentran de acuerdo con los diferentes contenidos de las diferentes columnas.

| Término | Significado |
|---|---|
| Afinidad (o Tabla o Archivo). | Tabla de dos dimensiones. |
| Atributo (o Columna o Campo). | Columna de una tabla. |
| Tupla (o Fila o Registro). | Renglón de una tabla. |
| Dominio. | Descripción física y lógica de los valores permitidos. |
| Llave. | Grupo de uno o más campos que identifican de modo único a un registro de una tabla. |
| Indice. | Grupo de uno o más campos soportados por una estructura de datos que facilita la recuperación y el acceso a los datos. |

**Resumen de la terminología relacional.**

Se pueden formalizar las tablas de datos aplicando una serie de reglas de normalización, con las que las tablas se organizan de una forma mejor para reducir la redundancia de los datos. La normalización, por tanto, da como resultado una base

de datos que refleja la naturaleza de los datos con simplicidad y con una pequeña redundancia de datos.

## NORMALIZACIÓN

La normalización es una técnica de diseño frecuentemente utilizada como una guía en el diseño de bases de datos relacionales. La normalización es esencialmente el proceso que consiste en poner los datos dentro de una forma tabular, removiendo la repetición de grupos y la duplicación de datos de las tablas relacionales.

La teoría de normalización esta basada en los conceptos de ***formas normales***. Una tabla relacional se dice que tiene una forma normal particular si esta satisface un conjunto de restricciones.

Inicialmente una tabla relacional por definición esta en la ***primera forma normal (1NF).*** Donde una instancia dada de un objeto de datos tiene un valor y solo uno para cada atributo. Todos los valores de las columnas son atómicos, (los elementos del dominio son indivisibles) esto es que contienen un valor no repetitivo.

**Relación en forma no normal**

| TITULO | AUTORES | FECHA |
|---|---|---|
| Anochecer | Asimov, Silverberg | 1991 |
| Fundamentos de Bases de Datos | Korth, Silberschatz | 1987 |

Para convertir esta relación no normal a la 1NF, hay que hacer que en cada celda tenga un único valor,

| TITULO | AUTORES | FECHA |
|---|---|---|
| Anochecer | Asimov | 1991 |
| Anochecer | Silverberg | 1991 |
| Fundamentos de Bases de Datos | Korth | 1987 |
| Fundamentos de Bases de Datos | Silberschatz | 1987 |

Pero en esta forma pueden existir redundancia de datos generando actualizaciones anormales que son problemas que surgen cuando la información es modificada, almacenada o borrada. Para eliminar tales anomalías se cambia el formato de la tabla, dividiéndola en dos o más tablas. Cuando se hace esto las nuevas tablas están en otra forma normal.

Un segundo paso es asegurar que en una tabla todas sus columnas que no son llave dependan por completo de la llave primaria, las columnas que no lo cumplan se colocan en otra tabla (Segunda forma normal. 2NF) eliminando redundancia.

| ID | ACTIVIDAD | CUOTA |
|---|---|---|
| 100 | Esquí | 200 |
| 100 | Golf | 65 |
| 150 | Natación | 50 |
| 175 | Squash | 50 |
| 175 | Natación | 50 |
| 200 | Natación | 50 |
| 200 | Golf | 65 |

| ID | ACTIVIDAD |
|---|---|
| 100 | Esquí |
| 150 | Natación |
| 175 | Squash |
| 200 | Natación |

| ACTIVIDAD | CUOTA |
|---|---|
| Esquí | 250 |
| Natación | 50 |
| Squash | 50 |

El tercer paso, es eliminar cualquier dependencia transitoria que ocurre cuando una columna no llave es dependiente de otras columnas no llave. (Tercera forma normal 3NF)

Se puede decir que una base de datos queda normalizada cuando todas las columnas no llave en cualquier tabla contengan información que sea solo referente a la llave primaria de cada una de las tablas.

En concreto, el propósito de una sistema de base de datos es crear, actualizar, eliminar y desplegar objetos en dirección a los usuarios, al mismo tiempo que proteger en todo momento la seguridad e integridad de la base de datos.

## DBMS (Data Base Management System)

Un sistema de administración de base de datos es un conjunto de programas especializados diseñados para describir, proteger, almacenar y accesar una base de datos y superar las limitaciones del procesamiento tradicional de archivos. Un DBMS (Data Base Management System, Sistema de Administración de Base de Datos), permite centralizar las decisiones de la organización y la integración de los datos. Si una base de datos, se diseña, implanta y mantiene en forma correcta, un DBMS puede ayudar a la organización a aumentar su habilidad para responder a las necesidades cambiantes de información.

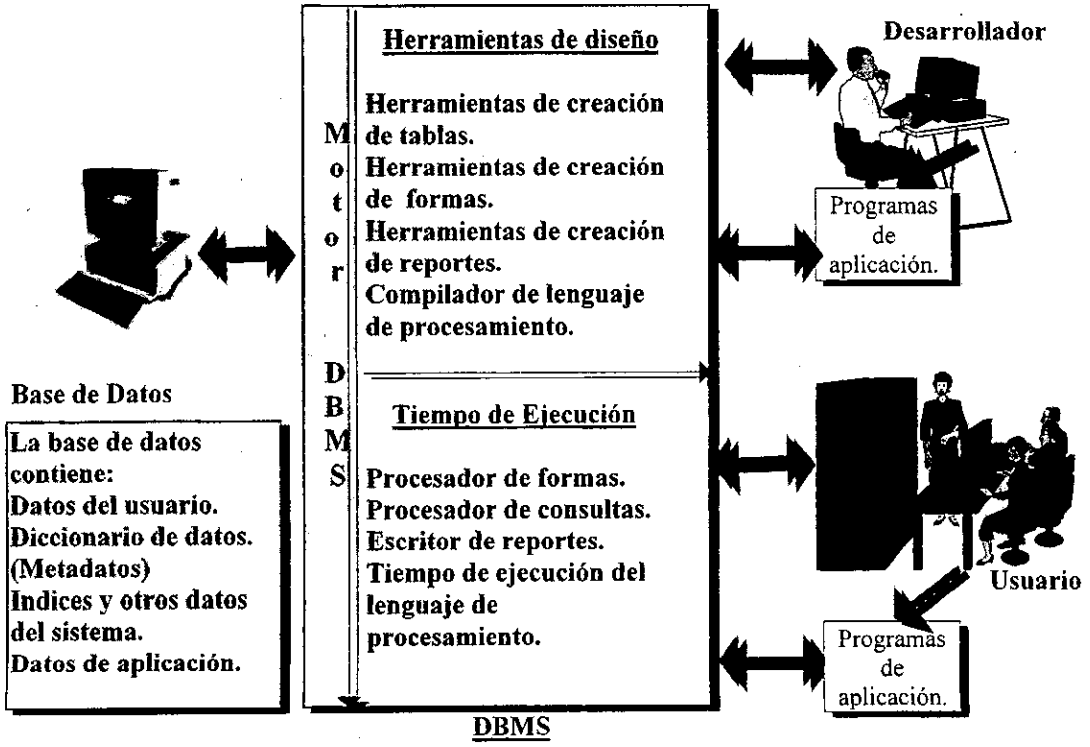
Un manejador o gestor de base de datos, es en sí, un conjunto de programas enfocados a conseguir como principal objetivo; funcionar como intermediario entre los datos (archivos) y los usuarios o programas. Algunas de las propiedades más útiles de los manejadores de bases de datos son:

- Independencia. Entre los datos y los programas. Si cambian los datos, no hay que corregir los programas.

- Visión global de la aplicación. Como normalmente se plantea, en un manejador se diseñan todos los archivos de datos que estarán involucrados en la aplicación. Esto permite reducir al máximo la repetición de datos.

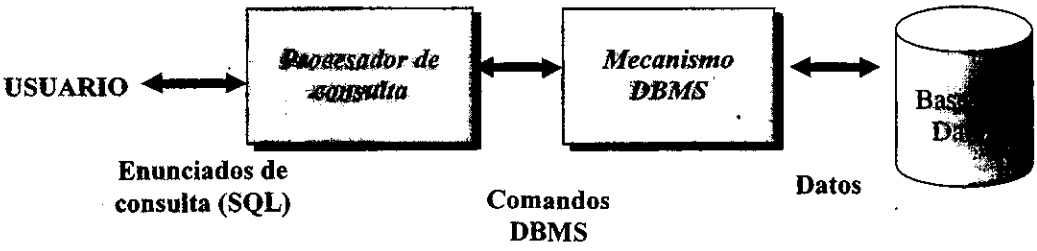
- Lenguaje de interface hacia el usuario. Los manejadores de base de datos tienen un lenguaje de consulta comúnmente llamado Query, el cual puede ser propio, apegado al estándar SQL ANSI.

Obsérvese la distinción entre Manejador de Base de datos y Bases de Datos en sí: los primeros son los programas que controlan y actualizan a las segundas.



Un sistema de administración de bases de datos (DBMS) es simplemente un software con el que se manejan los datos. Un sistema de administración de bases de datos relacionales (RDBMS) es un DBMS basado en el modelo relacional, que proporciona un mejoramiento en la disponibilidad de los datos, integridad y seguridad, porque el modelo esta rigurosamente definido para estas características como parte de la base de datos, no como parte del proceso de mantenimiento de la base de datos.

Además, un gestor relacional debe por tanto, proporcionar un lenguaje para operar con las tablas, con capacidad de manejo de conjuntos de filas tanto en lectura como en actualización, y con una potencia expresiva equivalente al cálculo relacional basado en predicados, en nuestro caso es SQL. Por lo que, todas las operaciones sobre tablas, se expresan en sentencias SQL, las que pueden estar embebidas dentro de programas o escritas directamente por usuarios terminales. Estas sentencias SQL son enviadas al DBMS para su análisis y proceso, el cual las revisa y las traduce a operaciones con los ficheros físicos, ejecuta éstas y devuelve el resultado al programa o usuario que hizo la petición. Puede haber varios de éstos trabajando simultáneamente con los mismos datos, tanto consultándolos como actualizándolos. El DBMS se responsabiliza de coordinar todas estas peticiones asegurando la integridad de los datos (para conocer la estructura de las sentencias SQL, ver apéndice A).




**Elementos que involucra el proceso de consulta.**

## CATÁLOGO DE SISTEMA

Un sistema de gestión de base de datos debe llevar la cuenta de gran cantidad de información referente a la estructura de una base de datos con el fin de efectuar sus funciones de gestión de datos. En una base de datos relacional, esta información está almacenada típicamente en el ***catálogo de sistema***, una colección de tablas del sistema que el DBMS mantiene para su propio uso. La información del catálogo describe las tablas, las vistas, las columnas, los privilegios y otras características estructurales de la base de datos.

El catálogo del sistema, por tanto es una colección de tablas especiales en una base de datos que son propiedad, están creadas y son mantenidas por el propio DBMS. Estas tablas son automáticamente creadas al crear la base de datos. Generalmente se almacenan todas juntas bajo un identificador tal como: SYSTEM, SYSIBM, MASTER o DBA.

El DBMS se refiere constantemente a los datos del catálogo de sistema cuando procesa las sentencias SQL. Por ejemplo, para procesar una sentencia SELECT de dos tablas el DBMS debe: verificar que las dos tablas existan realmente, asegurar que el usuario tiene permiso para acceder a ellas, comprobar que existen las columnas referenciadas en la consulta, determinar el tipo de datos de cada columna, entre otras cosas.

Almacenando la información estructural en tablas de sistema el DBMS puede utilizar sus propios métodos y lógica para recuperar rápida y eficientemente la información que necesita en la realización de sus tareas.

Es importante mencionar que las tablas pueden ser consultadas (sólo lectura) por los usuarios a través de consultas SQL estándar. Sin embargo, la capacidad de

actualizar o modificar directamente las tablas la tiene sólo el DBMS, ya que tales modificaciones por parte del usuario destruirían la integridad de la base de datos.

## TRANSACCIÓN, ROLLBACK Y BITACORAS.

Las bases de datos son herramientas que simplifican diversas tareas, entre sus tareas se encuentran la de recuperar información en caso de fallas, así como el asegurar la consistencia de dicha información, esto es, que la información siempre debe reflejar el estado del mundo real y que a pesar de la falla de algún proceso (por software o hardware) un grupo de actualizaciones a diversas tablas no se queden a medias. Para poder asegurar dicha consistencia, se creo el concepto de transacción, la cual es un bloque indivisible de actualizaciones a diversos archivos (tablas), en otras palabras una transacción sólo puede estar en dos estados: no hecha o hecha totalmente. Si algo llegará a suceder en el período entre el inicio y el fin de la transacción, ésta debe ser eliminada, para esto, el método más práctico es llevar una bitácora donde se grabe el estado anterior y el movimiento que afecta a la fila y a la columna específica, de esta forma si algo sucede durante la transacción, se recorrerá la bitácora y se volverán a colocar los datos en su estado inicial. A lo anterior se le denomina **Roll-Back** y a la bitácora es común que se le llame **LOG**.

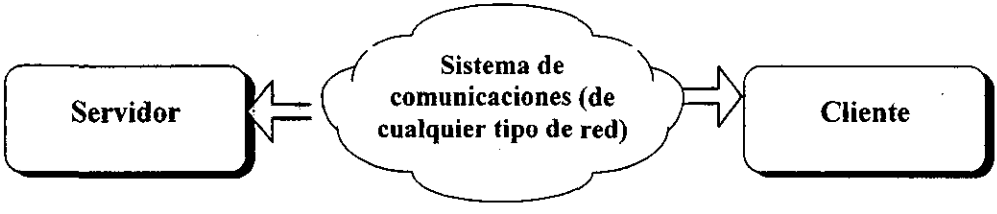
SQL es un vehículo natural para implementar aplicaciones utilizando una arquitectura cliente/servidor distribuida. En este papel, SQL sirve como enlace entre los sistemas informáticos (frontales) optimizados para la interacción con el usuario y los sistemas (de apoyo) especializados para gestión de base de datos, permitiendo que cada sistema rinda lo mejor posible.

# II.3 ARQUITECTURA CLIENTE / SERVIDOR

Típicamente una arquitectura cliente/servidor parte del proceso entre el cliente y el servidor sobre una red. El cliente presenta al usuario la interface, genera las peticiones y procesa dichos datos. El servidor responde a las peticiones del cliente y asegura la integridad de los datos.

La arquitectura cliente/servidor, esta formada por uno o más clientes y uno o más servidores, que junto con el sistema operativo y los protocolos de comunicación, conforman el ambiente que permite y facilita el computo distribuido[1].

La computadora central donde se ejecutan la mayoría de los sistemas operativos de red, se llama servidor. Una computadora que utilice recursos administrados por un servidor se conoce como cliente. Todas las computadoras de una red de este tipo se designan ya sea como servidores o como clientes: los servidores proporcionan servicios y los clientes utilizan esos servicios. La filosofía de la arquitectura cliente/servidor es tan simple - y tan compleja - como eso.



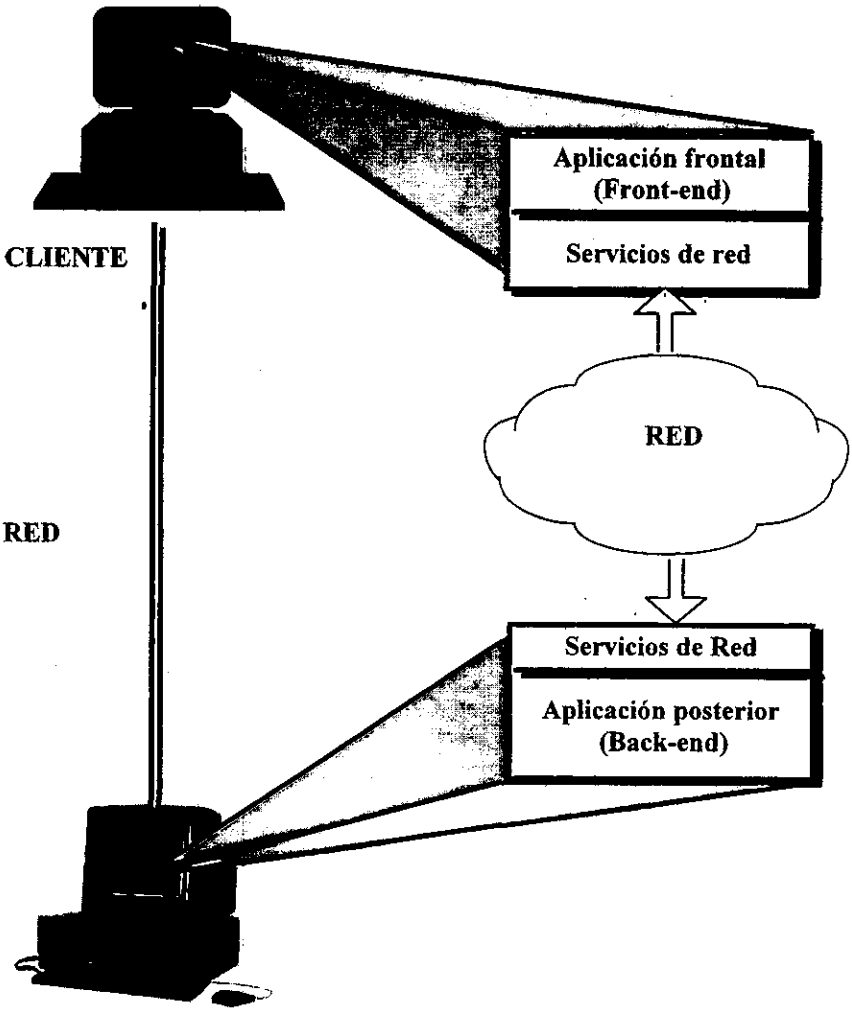

[1] **Computo Distribuido.** La idea o filosofía del computo distribuido nos permite utilizar todos los recursos de todas las máquinas de una manera corporativa y sin sobrecargar ni depender de un solo equipo.

De forma concisa, se puede decir que un servidor es un sistema o un programa que provee de algún servicio a otros sistemas a través de una red y un cliente es un sistema que requiere y recibe alguna acción de un servidor.

Un servidor por tanto es un administrador de recursos. Un recurso puede ser identificado como algo físico o abstracto, tal es el caso de una base de datos. Un cliente es un usuario de los recursos que administra el servidor.

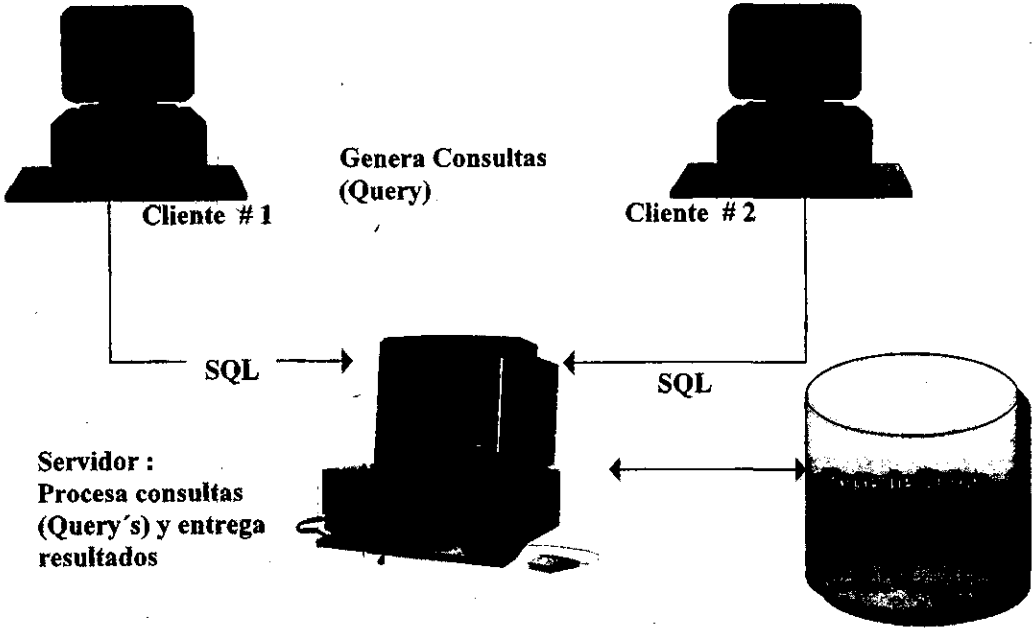
Bajo este esquema se reparte el proceso de una aplicación entre un ***front-end*** y un ***back-end.*** Donde la parte que corre en el servidor de base de datos se denomina back-end (tales como Sybase, Access, Informix, Progress, entre otros) y la que corre en el cliente se llama Front-end (Power Builder, Visual Basic, Visual C++, Delphi, etc.).

Un producto de aplicación frontal (***front-end***) es el software que se encarga de manejar las solicitudes del usuario y controlar el acceso a los servicios y recursos de la red. En un sistema de red, el producto de aplicación posterior (***back-end***) se encuentra en otra computadora, (aunque también puede estar en un módulo separado de la misma PC, cuando esta no esta bajo la arquitectura cliente/servidor). La red conecta la aplicación frontal con la aplicación posterior.




**Productos *front-end* y *back-end***

De manera general, para que se inicie la comunicación entre un cliente y un servidor es necesario establecer una sesión. Por lo tanto, el servidor debe estar esperando que algún cliente trate de establecer una sesión.




**Acceso a la base de datos Cliente/Servidor**

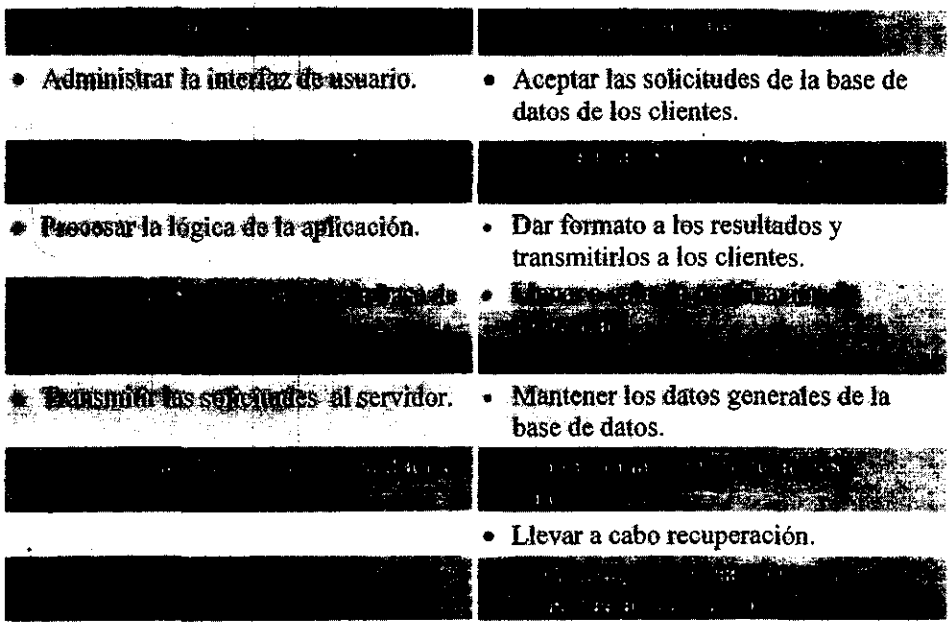
En la siguiente tabla se resumen las funciones tanto del cliente como del servidor, pero antes se mencionan elementos a considerar para ambos:

En cuanto al **Cliente**:

- Sistema Operativo de la Estación de Trabajo.- Se deben diseñar sistemas cliente/servidor que soporten diversos sistemas operativos en las estaciones de trabajo.
- Consideraciones con respecto al Hardware.- El cliente debe ser capaz de soportar la aplicación en cuanto a recursos que demande el ***front-end*** que maneje, para su ejecución optima.
- Consideraciones con respecto a la Conectividad.- Depende directamente de la red, (topología, protocolo, medio de comunicación, velocidad de accesos, etc.).

Por otra parte en el **Servidor:**

- *Escalabilidad.- Debe soportar números crecientes de clientes.*
- Demanda de recursos. Contemplar los requerimientos tanto de hardware como de software que deberá sustentar el servidor para que proporcione un servicio eficiente y de calidad.
- Interfaz con el servidor.- Es importante que las características del servidor (tanto en software como en hardware), sean transparentes a los clientes, excepto en lo que se refiere a la interfaz estándar de acceso a sus servicios (como SQL).




[2] ***Acceso Concurrente.*** *En muchos casos, los usuarios necesitan accesar a los mismos datos más o menos al* mismo tiempo, para evitar estos conflictos, la mayoría de los sistemas de bases de datos colocan "candados" temporales en ciertos bloques de datos (a nivel archivo, registro, o campo) para asegurar que no se harán otras modificaciones a estos datos mientras se encuentran en etapa de proceso.

## Ventajas y Desventajas de la arquitectura Cliente / Servidor

Entre las principales ventajas que presenta esta arquitectura son:

- **La Seguridad** es mayor porque los datos de la base de datos del servidor son accesados de manera directa. Sólo aquellos usuarios que tengan permiso, podrán ver los archivos de datos.

- Se puede **mejorar el rendimiento**, ya que el servidor que maneja el ***back-end*** puede proporcionar una mejor coordinación de usuarios múltiples, consecuentemente un mejor rendimiento. En caso de servidores de bases de datos, los clientes no tienen que leer todos los datos para encontrar los que quieren, estos hacen solicitudes al servidor y el sólo entrega los datos que necesitan.

- **Efectividad con relación al costo**, puesto que los clientes sólo necesitan el poder suficiente para ejecutar adecuadamente el ***front-end***.

En contra parte las desventajas serían:

- **Complejidad**. Los sistemas cliente/servidor generalmente no son fáciles de configurar ni manejar.

- **Requerimientos**. Para dar servicios a muchos usuarios, el servidor frecuentemente requiere ejecutarse en una computadora robusta ya que las aplicaciones tienden a ser grandes y complejas y generalmente requieren gran capacidad de procesamiento y demandan mucha memoria.

Así, una arquitectura Cliente/Servidor divide a la aplicación en procesos separados que corren en distintas máquinas enlazadas por una red. Es por ello que en el momento de diseño se dividen las tareas en subtareas a ser llevadas a cabo ya sea por el cliente o por el (los) servidor(es), teniendo como únicas limitantes las facilidades que ofrezca el sistema operativo de red, así como las reglas de la organización.

Entre más avanzado sea un sistema operativo de red, las aplicaciones serán más pequeñas y fáciles de desarrollar en menos tiempo. Además, el hecho de que en el servidor resida la información de la organización permite incrementar la seguridad, pues se establecen mejores controles de acceso a la misma.

En algún momento se criticó que era difícil el acceso a los sistemas Cliente/Servidor, pero con tecnologías como el estándar Open Data Base Conectivity (Conectividad de Bases de Datos Abiertas, ODBC por sus siglas en inglés) de Microsoft, es muy fácil su manejo. ODBC elimina todos los detalles complejos de manera que el acceso sea fácil y sencillo, además de dar soporte a consultas a diferentes bases de datos.

## ODBC (Open Data Base Conectivity)

Cada base de datos incluye su propio lenguaje o interfaz para la programación de aplicaciones API (Aplication Programming Interface). Al seleccionar una base de datos, se obliga a emplear la interfaz definida para esta, lo que dificulta el acceso a datos de otras bases de datos y la distribución por la red. Aunque existen facilidades para importar y exportar datos, es menos frecuente encontrar utilidades para mover o acceder a otros datos.

El objetivo de ODBC es el acceso transparente desde un cliente a un servidor de datos, haciendo posible acceder a datos almacenados en las principales bases de datos relacionales (Oracle, Sybase, Informix, etc.), no relacionales (dBase) y archivos de hojas de calculo, procesadores de texto, etc.

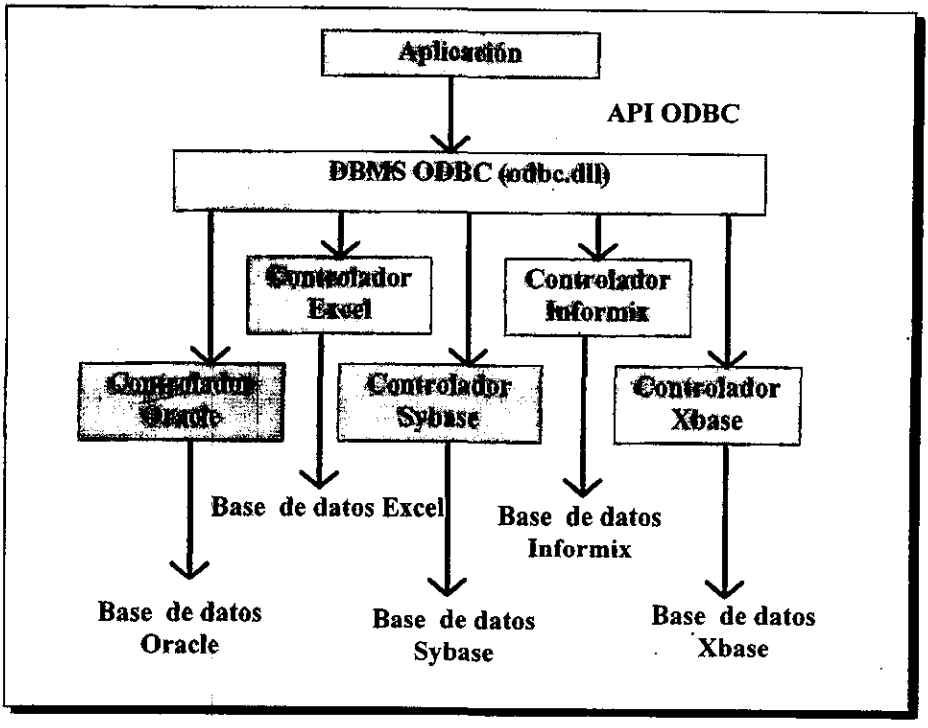
Con ODBC una aplicación puede escoger cualquier fuente de datos siempre que exista un controlador ODBC para la misma; es posible acceder simultáneamente a varia fuentes de datos.

El acceso transparente a los datos vía ODBC tiene lugar mediante el DBMS y los controladores específicos de cada base de datos. Una aplicación cliente llama al DBMS quien se encarga de cualquier intercambio de información entre la aplicación y la fuente de datos.

El DBMS procesa la llamada, valida su sintaxis, la convierte a la sintaxis de la base de datos destino y descarga los controladores ODBC conforme sean necesarios para ejecutar la llamada.

Una aplicación llamará al DBMS para comunicarse con más de un controlador a la vez, pues puede acceder a datos de distintas bases de datos, pero la API es la misma para todos. La aplicación no cambia si cambia la base de datos.

**Nota**: El controlador ODBC depende tanto de la aplicación y de la fuente de datos, como del sistema operativo base donde se ejecute.




Ahora bien, la forma de estructurar los procesos tanto por la interfaz de usuario, como con el DBMS precisan una metodología de programación que permita una eficiente ejecución de los procesos que conlleve el sistema.

## II.4 PROGRAMACION ORIENTADA A OBJETOS

La programación orientada a objetos (POO) es una de las más grandes ideas en el campo de la programación, donde todo se reduce a organizar sus programas en formas que reproducen la manera en que las cosas se unen en el mundo real. La

POO define los programas como un conjunto de estructuras de datos que tienen elementos de datos y conductas de programa.

La programación orientada a objetos funciona de la siguiente manera: al utilizarla, el programa general estará formado por componentes individuales (objetos) numerosos y diferentes; cada uno de los cuales realizará su papel en el programa y todos se comunicarán en formas predefinidas.

En esencia existen sólo cuatro componentes clave del lenguaje de POO:

- Objetos.
- Clases.
- Herencia
- Mensajes.

**Objeto**.

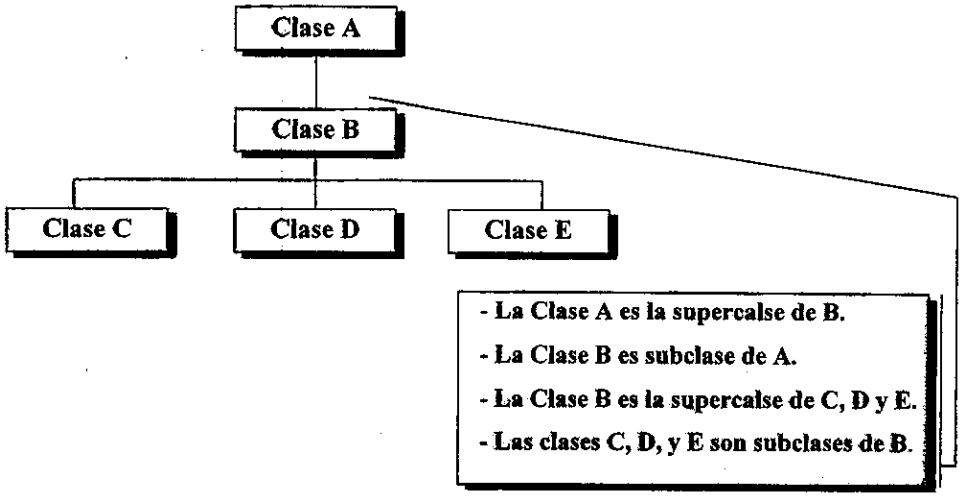
Definido de manera específica, un objeto de POO es una estructura encapsulada que tiene atributos (propiedades) y métodos. El término encapsulada significa que es completa por sí misma (una encapsulación se define como un reagrupamiento bajo un mismo nombre de datos y procedimientos que manejan solamente a estos datos y sólo a ellos); los programas ajenos a su estructura no conocen nada de su estructura ni necesitan saberlo. Por último, los objetos contienen métodos, los cuales son programas que los objetos de POO emplean para procesarse *a sí mismos.*

El acceso a los datos de un objeto es solamente a través de estos procedimientos; únicamente estos procedimientos pueden manipular, referenciar y/o modificar estos datos.

**Clase.**

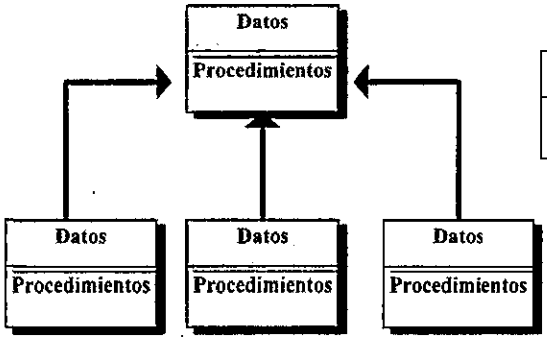
Para poder describir todos los objetos de un programa, conviene agrupar éstos en Clases. Se considera una clase como una colección de objetos que poseen características y operaciones comunes. Una clase contiene toda la información necesaria para crear nuevos objetos. Por lo que se define como un descriptor de propiedades que puede engendrar ejemplares utilizables denominados instancias u objetos. Por lo tanto y abreviando, un objeto o instancia es una entidad concreta que se fabrica o crea a partir de una clase. Existe un tipo peculiar de clase, denominado Metaclase, el cual es una clase donde sus instancias son a su vez clases.

Cada clase tiene una superclase (la clase superior en la jerarquía), y puede tener una o más subclases (las clases debajo de esa clase en la jerarquía). Se dice que las clases inferiores en la jerarquía, heredan de las clases más altas.

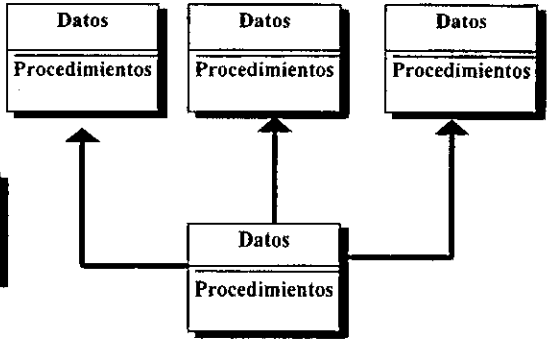


**Herencia**.

La herencia es la habilidad de una clase para definir la conducta y las estructuras de datos de las instancias como un superconjunto de la definición de otra clase o clases, en otras palabras, se puede decir, que una clase es igual a otra clase, excepto que la nueva clase incluye algo extra.




**Herencia Simple**
**Cada clase sólo puede tener un padre**




**Herencia Múltiple.**
**Cada clase puede tener múltiples padres**

Se pueden definir nuevas clases que contengan características heredadas de una o varias clases previamente definidas. Estas nuevas clases pueden ser restricciones o extensiones de una o varias clases anteriores, en cuyo caso la herencia será respectivamente simple o múltiple.

Las clases pueden tener más de una superclase, y heredar variables y métodos combinados de todas esas clase. A esto se le llama herencia múltiple.

La herencia nos permite concebir a una nueva clase de objetos como un refinamiento de la otra, al obtener las similitudes entre clases y al diseñar y especificar sólo las diferencias para la nueva clase. De esta forma podemos crear rápidamente clases. Así también se puede reusar código. Las clases que heredan de

otra clase pueden heredar su conducta, por lo que este mecanismo de abstracción de código puede proveer de un poderoso medio para producir código que puede ser reutilizado muchas veces.

**Mensajes.**

Los objetos de POO interactúan enviándose mensajes unos a otros. El método para modificar, envía un mensaje a otros objetos a fin de solicitar datos, efectuar modificaciones y obtener los datos necesarios. Los otros objetos reciben los mensajes y responden a ellos ejecutando sus propios métodos. Como todos los objetos están encapsulados, ninguno puede ni necesita conocer la estructura de otro objeto. Esto reduce la complejidad al mínimo y promueve una cohesión[3] eficiente.

Mediante el envió de mensajes se activan los puntos de entrada de las encapsulaciones. Al recibir el mensaje, la encapsulación ejecuta el procedimiento correspondiente. Una transmisión eficiente debe especificar lo siguiente:

- Nombre de la encapsulación que recibe el mensaje (receptor).
- Nombre del punto de entrada a activar (Selector del mensaje).
- Posibles argumentos eventuales.


[3] **Cohesión.** Es la característica propia de los objetos, refiriéndose a la unión de sus elementos, se mide en términos de la fuerza de unión de los elementos dentro de un objeto. Esta cohesión ocurre dentro de una escala, de la más débil (la menos deseada), a la más fuerte (la más deseada).

El mensaje enviado al receptor no indica nada de la estructura del receptor, únicamente conoce su nombre, esto permite modificar la representación interna, sin cambiar los programas llamantes.

Al recibir un mensaje una instancia de clase (objeto) se pueden considerar varios casos:

1. El mensaje corresponde a un procedimiento definido localmente en la clase del receptor. En este caso se realiza la ejecución del procedimiento.

2. El mensaje no corresponde a un punto de entrada de la clase receptor, si esta clase posee un padre, se envía el mismo mensaje al padre, y así recursivamente hasta encontrar por medio de la herencia la clase que contenga un punto de entrada correspondiente al selector.

3. El selector del mensaje no corresponde a ningún punto de entrada de la clase, ni tampoco al padre ni ascendentes. En este caso el envío del mensaje provoca un error.

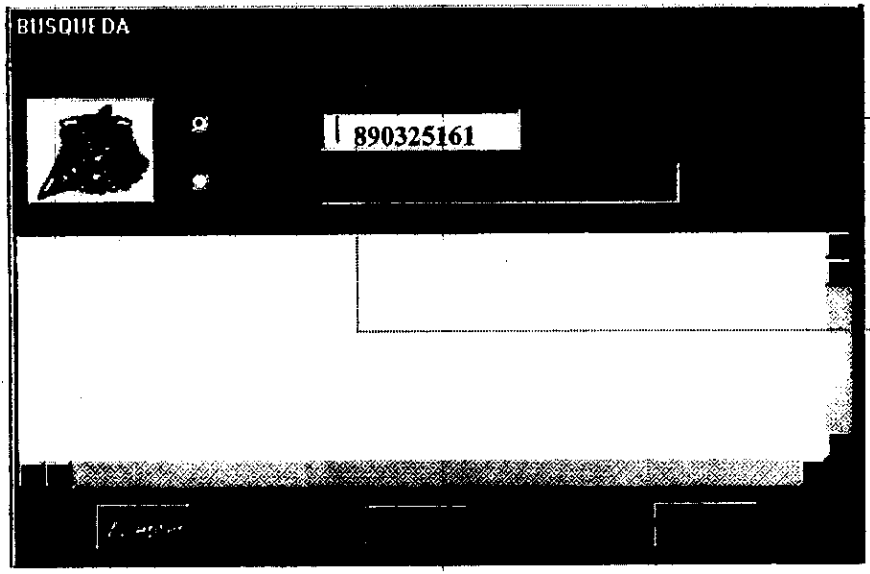
Los métodos en la POO funcionan mucho mejor por la característica llamada ***polimorfismo***, la cual permite implementar múltiples formas de un mismo método, dependiendo cada una de ellas de la clase sobre la que se realice la implantación. Esto hace que se pueda tener acceso a una variedad de métodos distintos (todos con el mismo nombre) utilizando exactamente el mismo medio de comunicación. Expresado en forma concisa, ***polimorfismo*** significa que pueden existir varias versiones de un método. De este modo se agrega funcionalidad sin invalidar propósitos ni operatibilidad anteriores.

Los objetos y los eventos de la programación están íntimamente relacionados, como sucede con los objetos y eventos de la vida real. Los eventos tienen lugar como resultado de la acción del usuario o del código del programa, o pueden ser activados por el sistema. La mayoría de los objetos responden a un número de eventos generados por el usuario, incluyendo un clic de ratón o doble clic, la opresión de una tecla o el arrastrar y soltar. En una aplicación visual tienen lugar la programación orientada a objetos manejada por eventos.

## II.5 PROGRAMACIÓN ORIENTADA A EVENTOS

La programación orientada a eventos permite, en lugar de escribir un programa que determina cada uno de los pasos en un orden determinado, diseñar un programa que responde a las acciones del usuario: elegir un comando, hacer clic en una ventana, mover el ratón, etc., en vez de escribir un gran programa, se crea una aplicación que es realmente una colección de microprogramas que cooperan entre ellos y que se ejecutan a raíz de eventos iniciados por el usuario.

En la programación orientada a eventos, la creación de un programa se realiza a partir de un grupo de objetos inteligentes que saben cómo responder cuando el usuario interactúa con ellos, después procesa la entrada utilizando procesamiento de eventos asociados con dichos objetos. Aun cuando la entrada pueda provenir del mismo sistema (una fecha especifica, un limite en algún dato, etc.). Independientemente de quien notifique un evento (suceso) el objeto reaccionará llamando al procedimiento del evento asociado a el.




La naturaleza dirigida por eventos, significa que la mayoría de los cálculos realizados en los programas se producirán por procedimientos de eventos. Estos bloques de código específicos de eventos procesan entradas, calculan nuevos valores, muestran la salida y gestionan otras tareas.

Cada objeto tiene un conjunto predefinido de sucesos a los que puede responder. Se puede escribir un procedimiento de evento para cualquiera de ellos, y si alguno de los eventos se produce en el programa, se ejecutará el procedimiento del evento asociado a el y debido a que los usuarios esperan que sus programas de computo se comporten de un modo confiable y predecible, es importante entender cómo la programación orientada a objetos y manejada por eventos contribuye a un formato bien estructurado y modular de su programa.

Pero, antes de la programación y estructuración de algoritmos, se debe contar con modelos que engloben y encaminen los procesos y todas y cada una de las funciones que abarcará el sistema, este proceso se denomina Diseño.

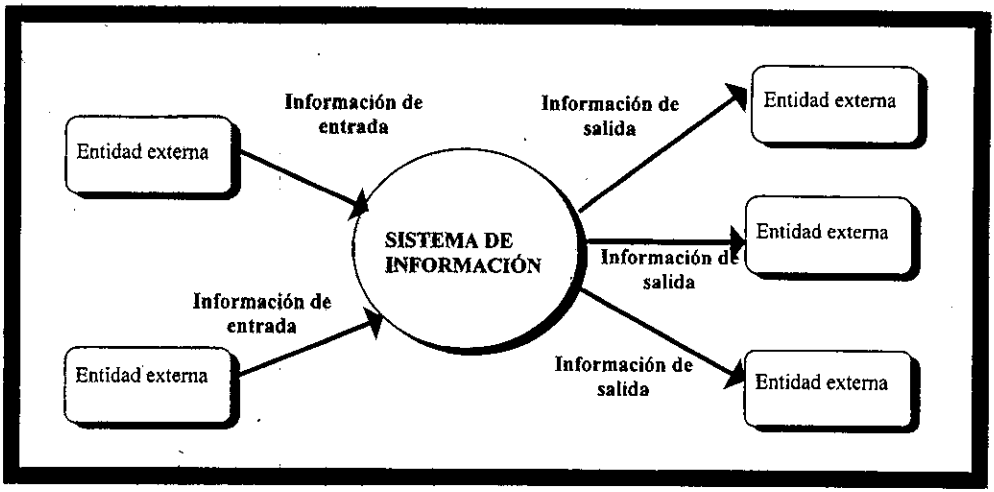
## II.6 DISEÑO DE SISTEMAS

El diseño general de sistemas puede definirse como el dibujo, planeación, bosquejo o arreglo de muchos elementos separados en un todo viable y unificado. En tanto que la fase del análisis de sistemas responde a las preguntas de lo que esta haciendo el sistema y de lo que debería estar haciendo para satisfacer los requerimientos de los usuarios, la fase del diseño se ocupa de la forma en que se desarrolla el sistema para satisfacer estos requerimientos.

A medida que se analizan los hechos de estudio y se revisan los requerimientos de usuario con otras aplicaciones familiares y similares, es conveniente formular y realizar modelos de diseño empleando algunas de las técnicas de modelación tales como: diagramas de flujo de datos y diagramas Entidad-Relación; ya sea en papel o en pantalla estos modelos se visualizan, se evalúan y se vuelven a dibujar hasta que parecen ser apropiados y factibles. En qué consisten y qué tipo de información proporcionan dichos diagramas, se verá a continuación.

## DIAGRAMA DE FLUJO DE DATOS.

A medida que la información se mueve a través del software, es modificada por una serie de transformaciones, el ***diagrama de flujo de datos*** **(DFD),** 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.

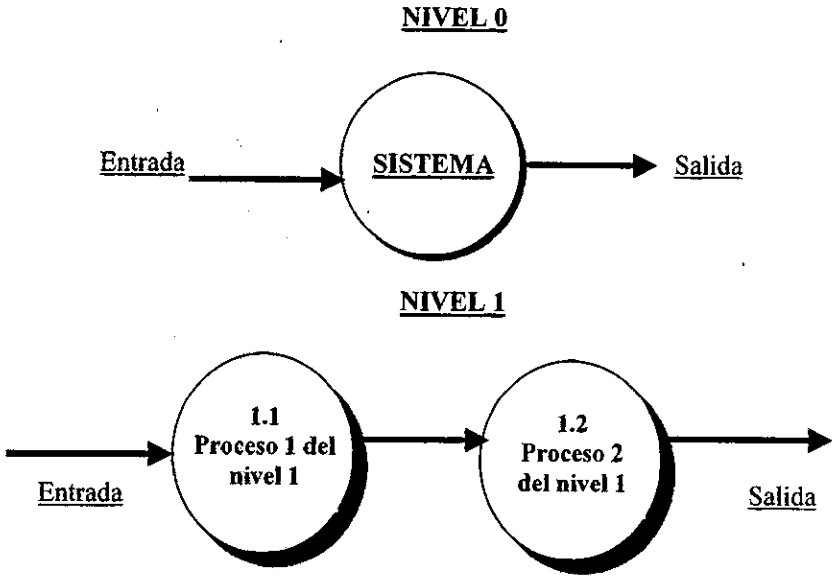



**Modelo de flujo de información**

Un Diagrama de flujo de datos puede representar un sistema en cualquier nivel de abstracción, es decir puede ir representando niveles con un mayor flujo de información y un mayor detalle funcional.

El **nivel 0** es la base y representa al sistema como un sólo proceso con datos de entrada y de salida. El **nivel 1**, abarcará por consiguiente los procesos principales y su interconexión, los niveles subsecuentes son derivados de estos, y se llegará hasta donde sea necesario. Es recomendable ayudarse de una notación única que

identifique cada proceso, generalmente se enumera secuencialmente el proceso, previo el nivel y separado por un ( . ), ejemplos:




El diagrama de flujo de datos DFD permite desarrollar los modelos del ámbito de información y del ámbito funcional al mismo tiempo. A medida que se refina el DFD en mayores niveles de detalle, se lleva a cabo implícitamente una descomposición funcional del sistema. al mismo tiempo, esto produce un nuevo refinamiento de los datos a la vez que se mueven a través de los procesos que componen la aplicación.

Sin embargo, es evidente que se requiere disponer de algún método que permita representar el contenido de cada componente del modelo de flujo, y aquel que lo proporciona es el diccionario de datos.

# DICCIONARIO DE DATOS

El diccionario de datos (también conocido como diccionario de requisitos, e incluso como ***metadatos***), describe el contenido de los objetos definidos en una base de datos.

El diccionario de datos es un listado organizado de todos los elementos de datos que son pertinentes para el sistema, con definiciones precisas y rigurosas que permiten al usuario y al creador del sistema, tener una misma comprensión de las entradas y salidas, de los componentes y también de los cálculos intermedios.

Básicamente contiene la siguiente información:

- **Nombre**. El nombre principal del elemento de datos o de control, de almacén de datos (archivos) o de una entidad externa.
- **Alias.** Otros nombres usados para las entradas.
- **Descripción del Contenido**. El contenido representado mediante una notación.
- **Información Adicional**. Información sobre los tipos de datos, los valores implícitos (sí se conocen), las restricciones o limitaciones, etc.

El diccionario de datos se considera como una librería o catálogo central, para describir y definir el significado, uso y características de los datos que sean relevantes como: campos, tipos de datos, sinónimos, alias, referencias cruzadas y las relaciones que existen entre todos ellos. Por lo que se le considera una buena herramienta de documentación. Entendiendo que el diccionario de datos no especifica valores actuales de los datos sino que define el tipo de valor que debe ir en el campo. Puede ser elaborado ya sea de forma manual o con la ayuda de herramientas de diseño de manera automatizada.

Finalmente se concluye que un diccionario de datos debe proporcionar la siguiente información:

- Listado de atributos y entidades.
- Listado de procesos.
- Verificación de referencias cruzadas. Cuando unos mismos datos se utilizan en diferentes procesos.
- Detección de errores.

Concretando se puede decir que debe contar con ciertas características, tales como: lógica, accesible por nombre, limitado, conciso, referido a un DFD y primordialmente comunicar significados.

## DIAGRAMA ENTIDAD - RELACION (E-R)

Un diagrama de Entidad-Relación (E-R) tiene tres elementos: Entidades, Relaciones y Atributos. Una entidad es una persona, lugar, objeto o concepto acerca del cual se almacenan datos. El principal propósito del diagrama E-R es representar los objetos de datos y sus relaciones. El modelo E-R, es un modelo conceptual, independientemente de máquinas o restricciones físicas.

La notación del diagrama es relativamente sencilla. Los objetos de datos se representan como rectángulos etiquetados. Las relaciones se indican mediante rombos. Las conexiones entre los objetos de datos se establecen mediante variadas líneas de conexión.

Las entidades son similares a nombres y los atributos corresponden a adjetivos, las relaciones similares a los verbos. Sólo las entidades tienen atributos.

La técnica de diagramas E-R que utiliza estos elementos permite asignar atributos a las entidades y definir relaciones entre las entidades dando por resultado un esquema de cualquier sistema que se entiende de manera fácil y sencilla.

Entre las entidades pueden existir relaciones y asociaciones, que pueden ser del tipo (1:1), (1:N) o (N:M), según sea la cardinalidad de la asociación, una cardinalidad 1 significa que a cada ocurrencia de una entidad le corresponde otra y solo esa de la entidad asociada, y una cardinalidad múltiple representada por N o M indica que a una ocurrencia de una entidad le corresponden varias de la entidad asociada. (en las bases de datos relacionales no se utiliza la relación N:M)

La notación E-R proporciona un mecanismo para representar la asociatividad entre los objetos, por lo que se convierte en una notación concisa para examinar los datos dentro del contexto de la aplicación de procesamiento de datos.

Sin embargo y para hacer más sencillo el manejo del diagrama, algunos autores recomiendan (observación de gran utilidad) el colocar las entidades con sus respectivos componentes y la relación conjuntarla con la llave primaria de estas, en cuanto a las bases de datos relacionales, con lo que se logra obtener de forma clara un entendimiento adicional sobre los detalles de los almacenes de datos (archivos) y sus relaciones con los procesos dentro del modelo de flujo, además de complementar la representación del contenido de los datos que se encuentran definidos en el diccionario de datos, encapsulando en un diagrama los datos y sus respectivas relaciones en todo el sistema.

Sintetizando, el análisis de un sistema que se basa en un modelo de flujo de datos como primer elemento de representación gráfica, permite separar las funciones que transforman el flujo y realizando un modelo del contenido de datos mediante un

diccionario de datos, se logra obtener una visión completa y correcta del mismo, permitiendo un desarrollo eficiente y de calidad en su base.

# CAPITULO III.

# ANÁLISIS Y DISEÑO

El proceso de desarrollo de sistemas consta de tres fases, la definición y análisis, el desarrollo y el mantenimiento; independientemente de la aplicación y/o el tamaño de este.

## III.1 ANÁLISIS DEL SISTEMA.

La fase del análisis del sistema define el papel de cada elemento del mismo, es decir se centra en el ***que***, en que información ha de ser procesada, que función y rendimiento se desea, que interfaces han de establecerse, que restricciones de diseño existen, que formatos de información se manejan, que periodicidad de reportes requiere, que criterios de validación necesita para definir el sistema correctamente, etc. Por lo que se deberán definir los requisitos clave del sistema, además de conformar un análisis de estos, proyectando una solución que satisfaga de manera óptima estos requisitos (proceso expuesto en el capítulo I).

Inicialmente, se estudian las especificaciones del sistema, los procedimientos que debe realizar, las funciones especificas que deberá cubrir, y sobre todo abarcar los tipos de salidas y reportes que demanden.

Al diseñar un sistema, en el análisis se deben hacer las siguientes consideraciones: requerimientos de salida, requerimientos de entrada, acceso, organización y almacenamiento de datos, procesamiento, controles del sistema, personal y procedimientos.

Las consideraciones de salida incluyen determinar la información que los usuarios necesitan, el nivel de detalle requerido, cuándo se necesitan los resultados, los medios, los dispositivos y el formato de los resultados de salida. Las consideraciones de entrada son similares a las de salida.

La salida es la información que reciben los usuarios del sistema, pero está relacionada con el contenido, los formatos, el tiempo de entrega y la frecuencia de actualización de la información proporcionada por el sistema, sin embargo, antes de convertirse en una salida adecuada, ciertos datos requieren de un proceso extensivo, otros sólo se almacenan y cuando se les solicita se consideran salidas con poco o nada de proceso. Las salidas pueden tomar diversas formas: en reportes impresos, en formatos establecidos, en información en pantalla, etc. Los usuarios confían en las salidas para la realización de sus tareas; y con frecuencia juzgan el mérito del sistema exclusivamente basándose en sus salidas. Durante la fase del análisis de determinación de requerimientos, se identifican los objetivos a cubrir y a partir de estos se diseñan las salidas.

En el diseño de la salida, se busca satisfacer los siguientes objetivos: la salida debe ser la adecuada para el usuario, proporcionar la cantidad correcta, presentarla

en el lugar adecuado, proporcionar la salida con oportunidad y elegir el método de salida más conveniente.

Los reportes impresos se diseñan a partir de los formatos y políticas establecidas en la institución, respetando la uniformidad de estos. El diccionario de datos sirve como fuente de los datos necesarios de cada reporte. Los bocetos y prototipos de reporte se presentan previamente para hacer cualquier cambio pertinente.

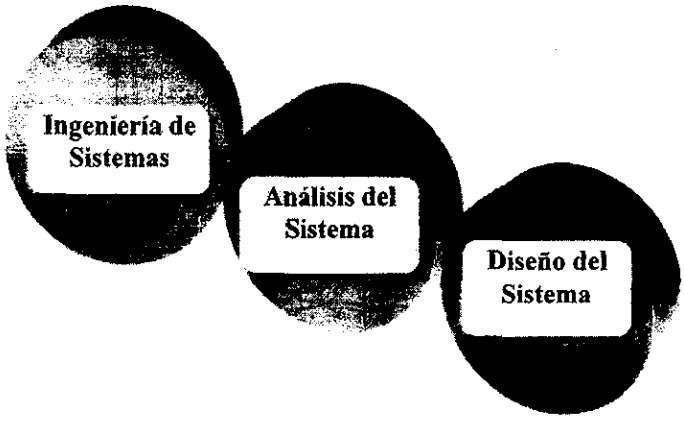
SABZ cuenta con una gran variedad de reportes que son tanto establecidos en formatos y periodicidad, como aquellos que son requeridos en lapsos aleatorios e indeterminados entre cada solicitud de estos, por lo que se debe diseñar el sistema de forma tal que cubra satisfactoriamente estas dos modalidades antagónicas. Y sobre todo que las salidas se presenten en la forma adecuada, a la persona que las requieren, en el tiempo justo.

Las consideraciones sobre acceso, organización y almacenamiento de datos incluyen la determinación de los tipos de acceso a los datos por parte de los usuarios, la organización de los elementos de datos para permitir estas clases de acceso y la elección de los medios y dispositivos de almacenamiento adecuados.

El diseño de la entrada especifica cómo se introducen los datos al sistema para su procesamiento, incluyendo métodos para la captación de datos y validación de su llenado completo, los objetivos básicos son el reducir la cantidad de datos para la entrada, además de controlar los errores y la demora. Evitando pasos extras, asegurando que el proceso en su totalidad sea lo más sencillo para el usuario y el personal que introduce los datos.

Sin importar el método de procesamiento o de entrada de datos, la entrada debe ser correcta. Debe certificar el llenado completo, asegurando la secuencia y que los datos no falten, así también que sean consistentes y que se encuentren dentro de los límites. Esto se realiza con el fin de encontrar errores antes de procesar y almacenar los datos.

Por lo que el análisis del sistema es la etapa que une la asignación del Sistemas con el diseño del sistema. El análisis de requisitos facilita la especificación de la función y el rendimiento del sistema, la descripción de la interfaz con otros elementos y el establecimiento de las restricciones de diseño que se deben considerar.

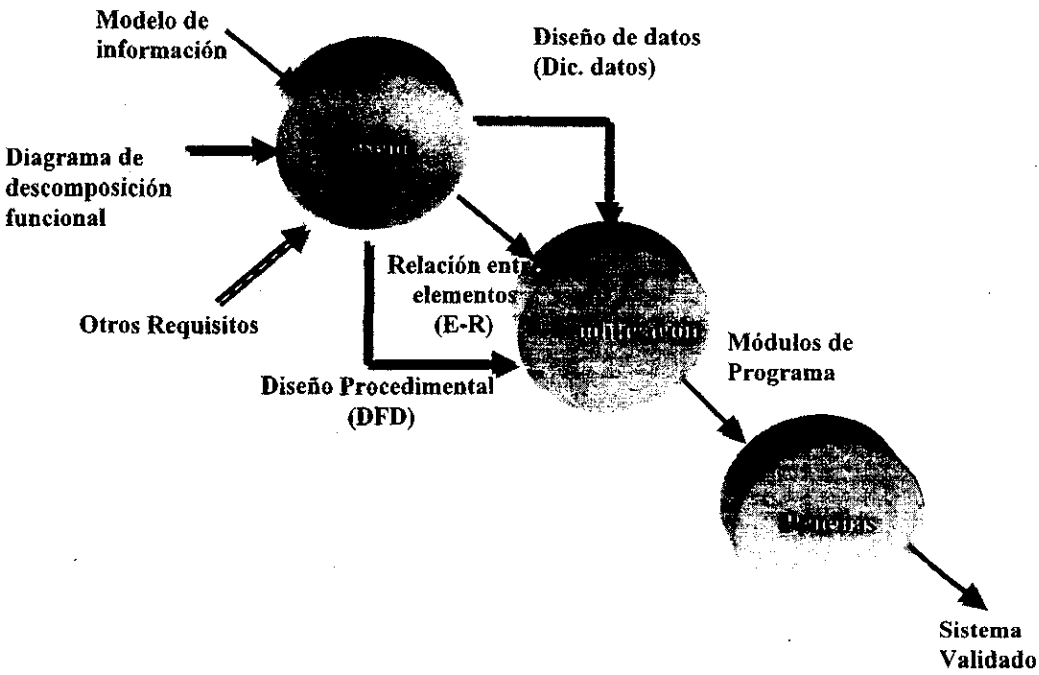



La *descripción del ámbito de la información contenida en las* especificaciones de requisitos del sistema, perfila la arquitectura que ha de desarrollarse durante el diseño.

## III.2 DISEÑO DEL SISTEMA.

Como se ha expuesto, la fase del análisis se basa en el ***que,*** y por otra parte la fase de desarrollo se centra en el ***cómo***, esto es, en descubrir cómo han de diseñarse las estructuras de datos y la arquitectura del sistema, cómo se han de implantar los detalles procedimentales y el cómo se traducirá el diseño a un lenguaje de programación, así también del cómo han de realizarse las pruebas.

Una vez que se han establecido los requisitos del sistema, el diseño del software es la primera de las tres actividades técnicas: diseño, codificación y pruebas. Cada actividad transforma la información de forma que finalmente se obtiene un sistema para computadora valido.



Los requisitos del sistema, establecidos mediante los modelos funcionales (diagrama de descomposición funcional), de información (entrevistas, observación y análisis) y de comportamiento (diagrama de contexto y procesos actuales del sistema existente y/o realizados manualmente), alimentan el paso del diseño. Mediante alguna metodología de diseño (diagramas, diccionario de datos, etc.) se realiza el diseño de datos, diseño de flujo de procesos, y la forma en que se relacionan entre sí cada elemento que lo conformará. El diseño de datos transforma el modelo de información en las estructuras de datos que se van a requerir para implantar el sistema. El diseño de flujo de procesos transforma los elementos estructurales en una descripción procedimental del sistema. Y finalmente se definen las relaciones entre los principales componentes que integran el sistema. Se genera el código fuente, y para integrar y validar el sistema, se llevan a cabo las pruebas.

El diseño de un sistema debe cumplir con las siguientes características:

1. Un diseño debe exhibir una organización jerárquica que haga un uso inteligente del control entre los componentes del sistema.
2. El diseño debe ser modular, es decir, debe estar dividido en forma lógica en elementos que realicen funciones y subfunciones especificas.
3. Debe contar con representación de los datos separada de los procedimientos.

El diseño del sistema consiste en la traducción de los requisitos de este a un conjunto de representaciones (gráficas o tabulares) que describan la estructura de los datos, la arquitectura, el procedimiento algorítmico y las características de la interfaz. La codificación por su parte abarca la traducción que se debe realizar a un lenguaje de programación. Y una vez que el sistema ha sido implantado se debe probar, para descubrir los defectos que puedan existir en el proceso, en la lógica y en la implantación.

# DISEÑO DE SABZ.

Las representaciones y/o herramientas en las que se apoyo el desarrollo del sistema SABZ son: El diagrama de flujo, El diccionario de datos y El diagrama Entidad-Relación, obteniendo lo siguiente:

## Diagramas de flujo de Datos.

La técnica denominada DFD (Diagrama de Flujo de Datos) proporciona una representación gráfica de cómo los datos se mueven a través de la organización, los procesos o transformaciones que los datos experimentan y el tipo de salida que presentan. El DFD permite enfatizar la lógica del sistema, usando combinaciones de sólo 4 símbolos. Los elementos para conformar un DFD conocido también como diagrama de burbuja son:

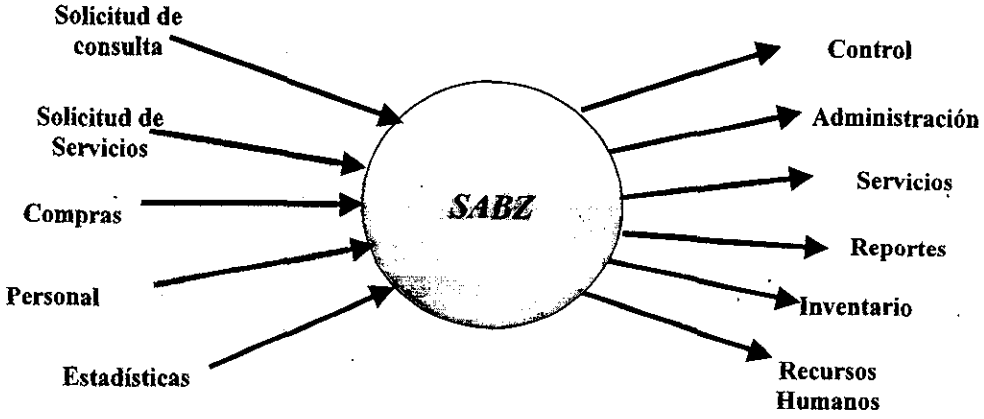
| | |
|---|---|
| **Unidad Externa** | Un producto o consumidor de información que reside fuera de los límites del sistema a ser modelado. |
| **Proceso** | Un transformador de información que reside dentro de los límites del sistema modelado |
| | Un elemento de datos o una colección de elementos de datos; la cabeza de la flecha indica la dirección del flujo de datos. |
| **Almacén de Datos** | Un depósito de datos que se guardan para ser usado por uno o más procesos; puede ser tan sencillo como un buffer o una cola, o tan sofisticado como una base de datos relacional. |

**Notación básica de un DFD**

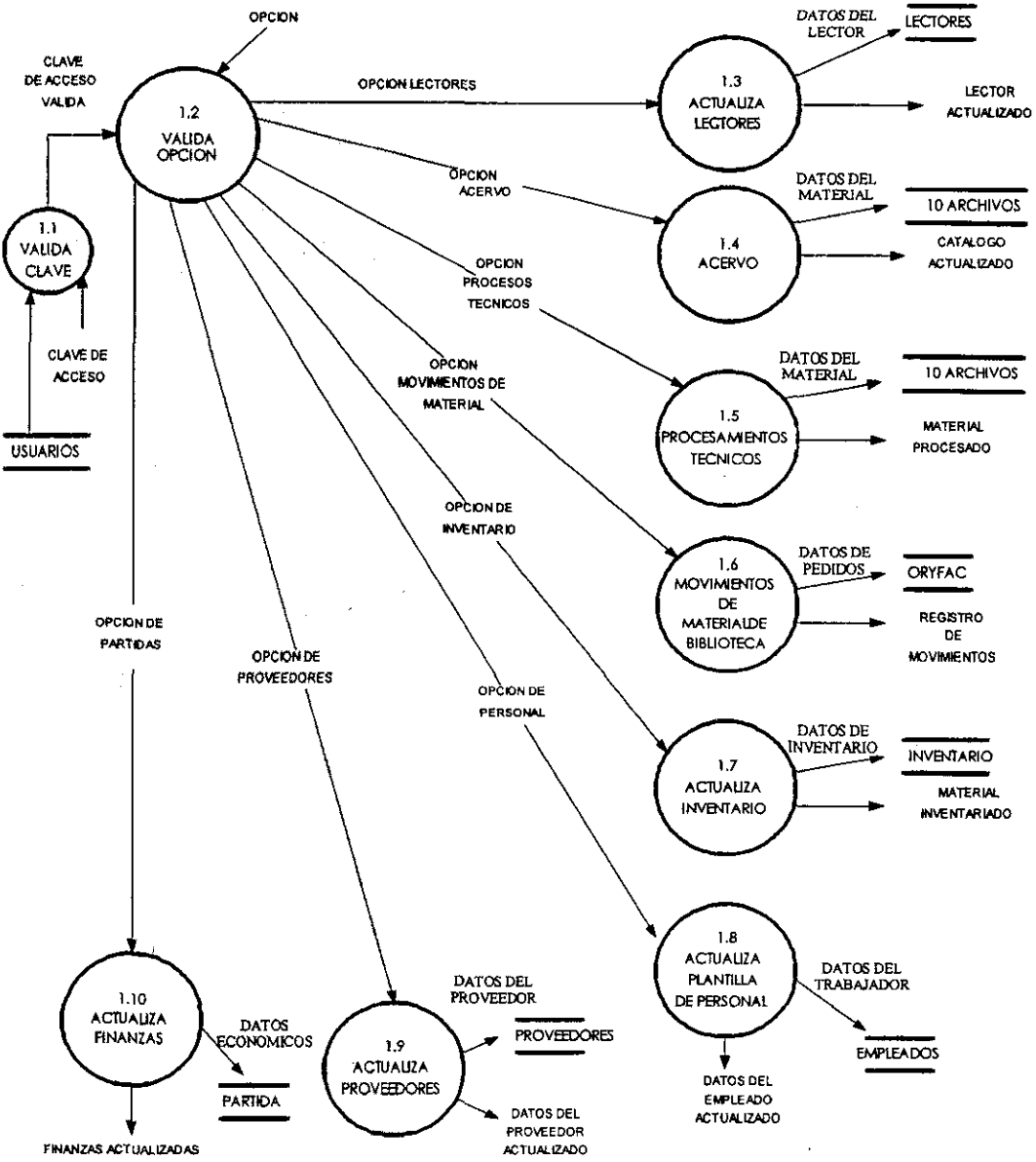
Los principales diagramas de flujo de datos de SABZ y que nacen a partir de un diagrama de descomposición funcional se presentan a continuación:

**SABZ**

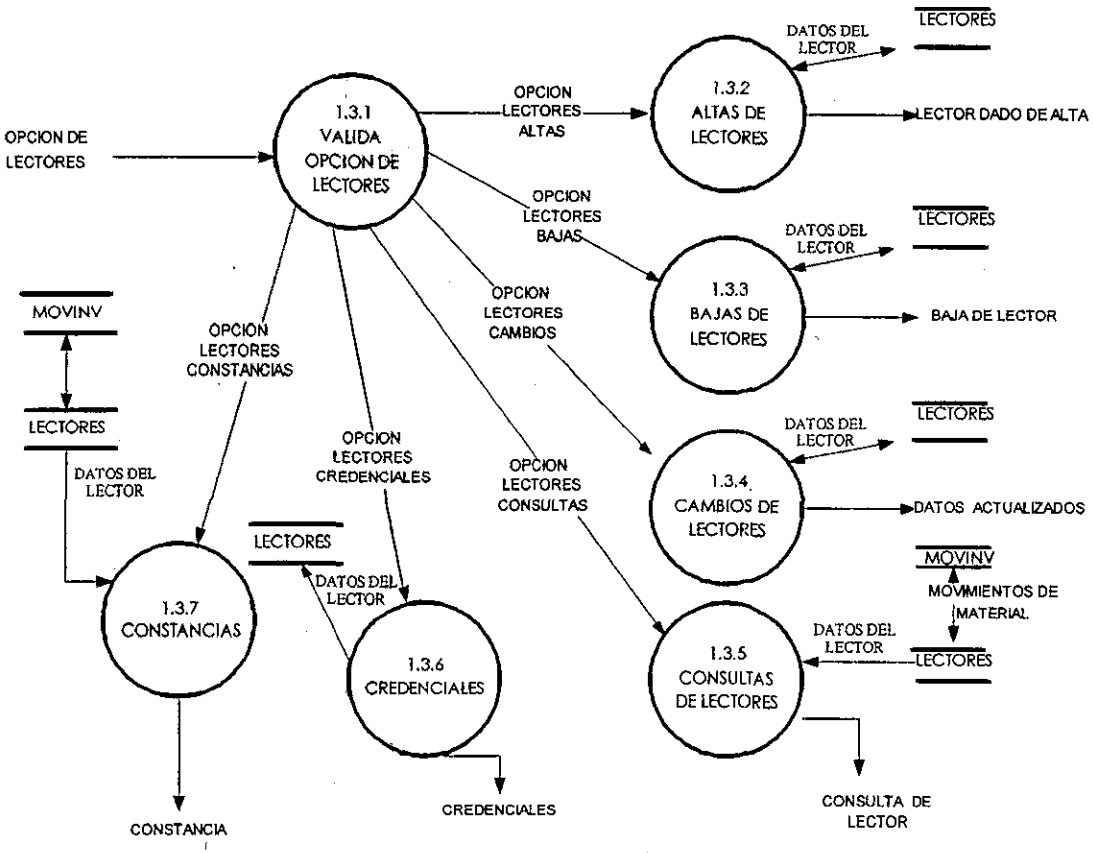
**NIVEL 0**

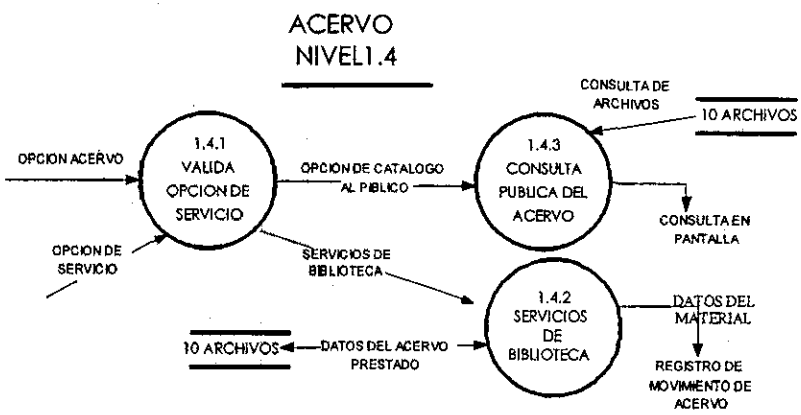


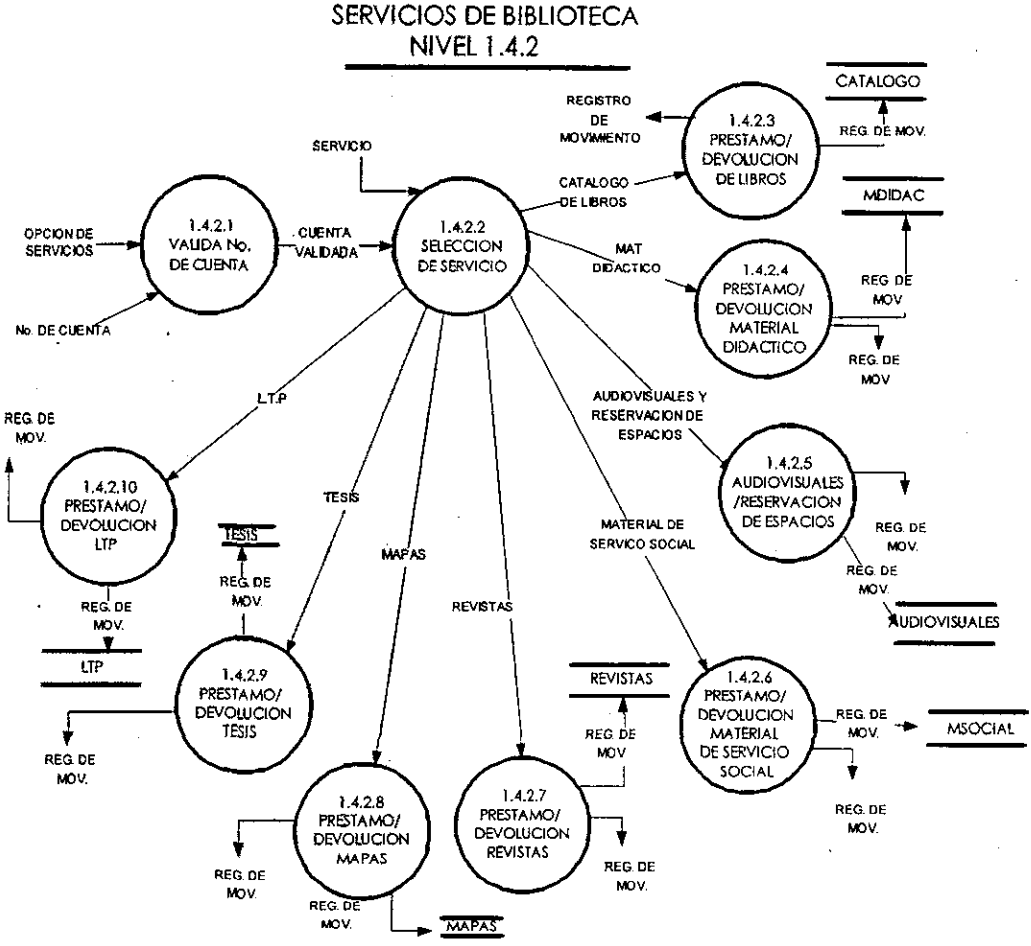
## ADMINISTRACION
## NIVEL 1



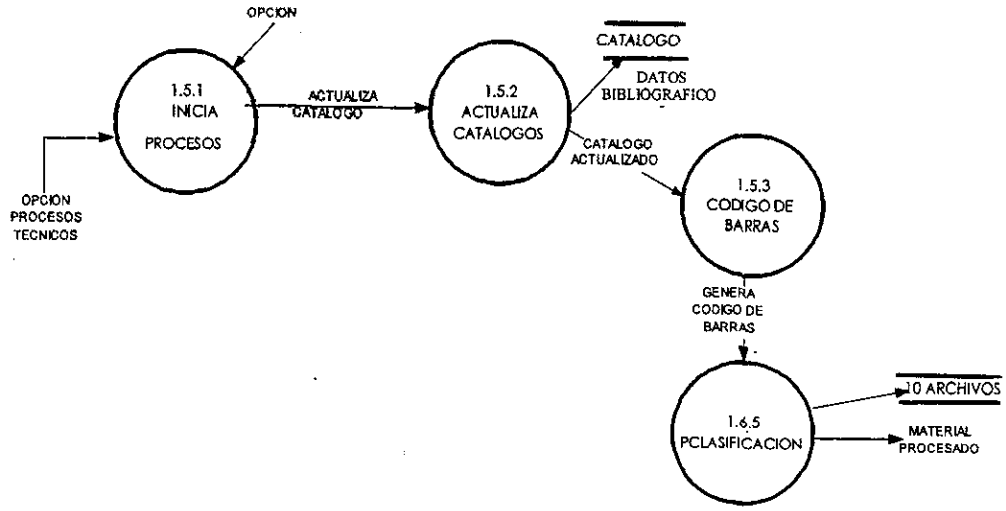
## ACTUALIZA LECTORES
## NIVEL 1.3



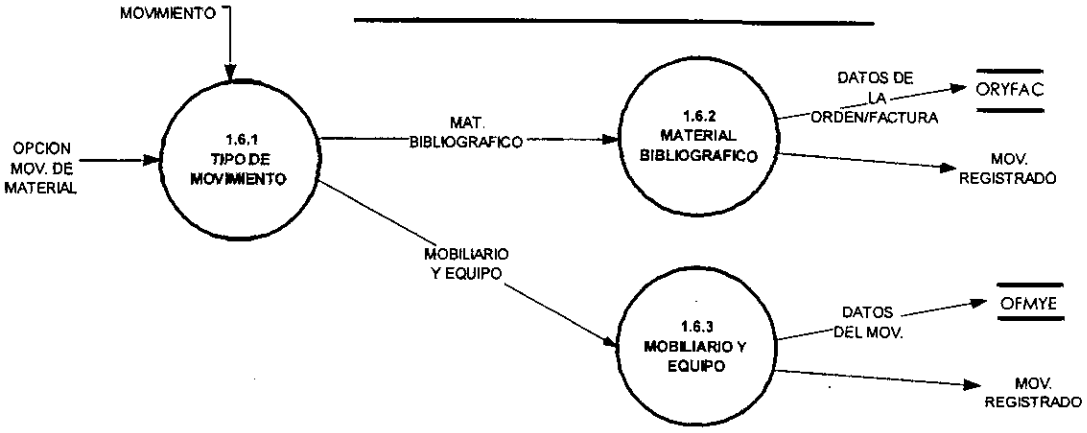

ACERVO
NIVEL1.4
CONSULTA DE ARCHIVOS
10 ARCHIVOS
OPCION ACERVO
1.4.1 VALIDA OPCION DE SERVICIO
OPCION DE CATALOGO AL PUBLICO
1.4.3 CONSULTA PUBLICA DEL ACERVO
CONSULTA EN PANTALLA
OPCION DE SERVICIO
SERVICIOS DE BIBLIOTECA
1.4.2 SERVICIOS DE BIBLIOTECA
DATOS DEL MATERIAL
10 ARCHIVOS
DATOS DEL ACERVO PRESTADO
REGISTRO DE MOVIMIENTO DE ACERVO

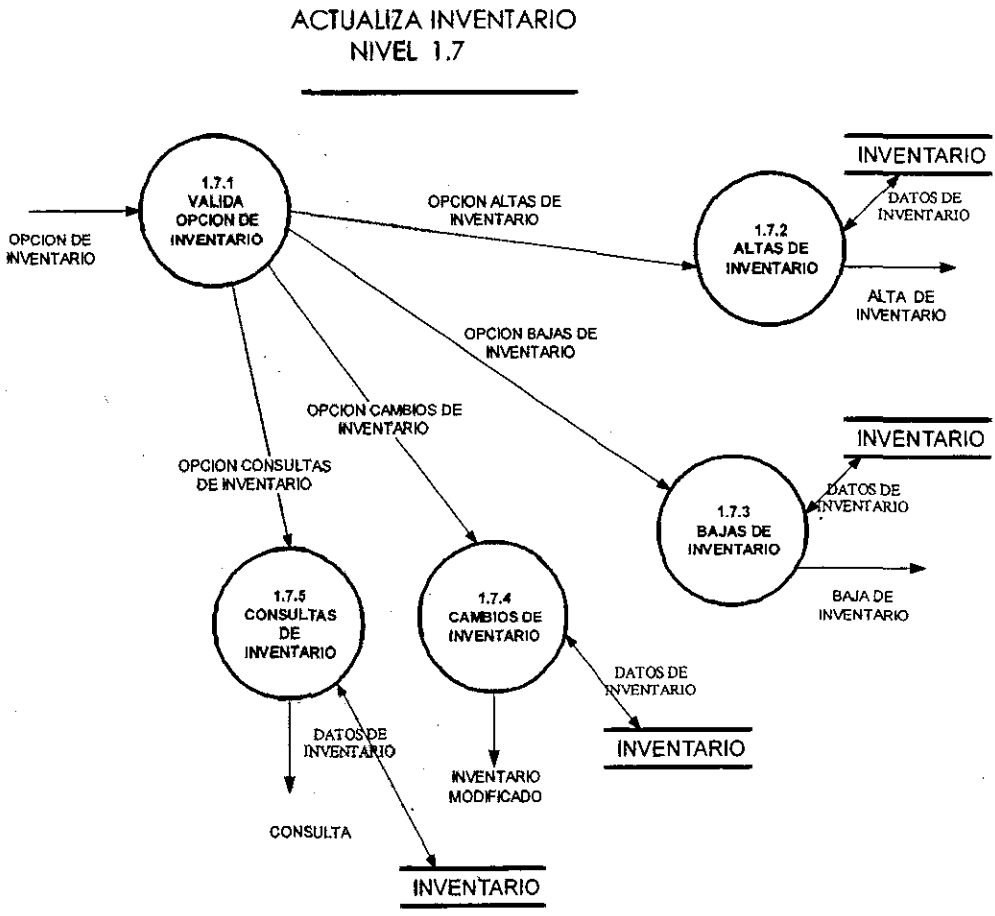

SERVICIOS DE BIBLIOTECA
NIVEL 1.4.2
CATALOGO
REGISTRO DE MOVIMIENTO
1.4.2.3 PRESTAMO/ DEVOLUCION DE LIBROS
REG. DE MOV.
SERVICIO
CATALOGO DE LIBROS
MDIDAC
OPCION DE SERVICIOS
1.4.2.1 VALIDA No. DE CUENTA
CUENTA VALIDADA
1.4.2.2 SELECCION DE SERVICIO
MAT DIDACTICO
1.4.2.4 PRESTAMO/ DEVOLUCION MATERIAL DIDACTICO
REG. DE MOV
No. DE CUENTA
REG. DE MOV.
L.T.P
AUDIOVISUALES Y RESERVACION DE ESPACIOS
REG. DE MOV.
1.4.2.10 PRESTAMO/ DEVOLUCION LTP
1.4.2.5 AUDIOVISUALES /RESERVACION DE ESPACIOS
TESIS
MAPAS
MATERIAL DE SERVICO SOCIAL
REG. DE MOV.
REG. DE MOV.
AUDIOVISUALES
REG. DE MOV.
REVISTAS
LTP
1.4.2.9 PRESTAMO/ DEVOLUCION TESIS
REVISTAS
1.4.2.6 PRESTAMO/ DEVOLUCION MATERIAL DE SERVICIO SOCIAL
REG. DE MOV.
MSOCIAL
REG. DE MOV.
REG. DE MOV.
1.4.2.8 PRESTAMO/ DEVOLUCION MAPAS
1.4.2.7 PRESTAMO/ DEVOLUCION REVISTAS
REG. DE MOV.
REG. DE MOV.
REG. DE MOV.
MAPAS

## PROCESAMIENTOS TECNICOS
## NIVEL 1.5

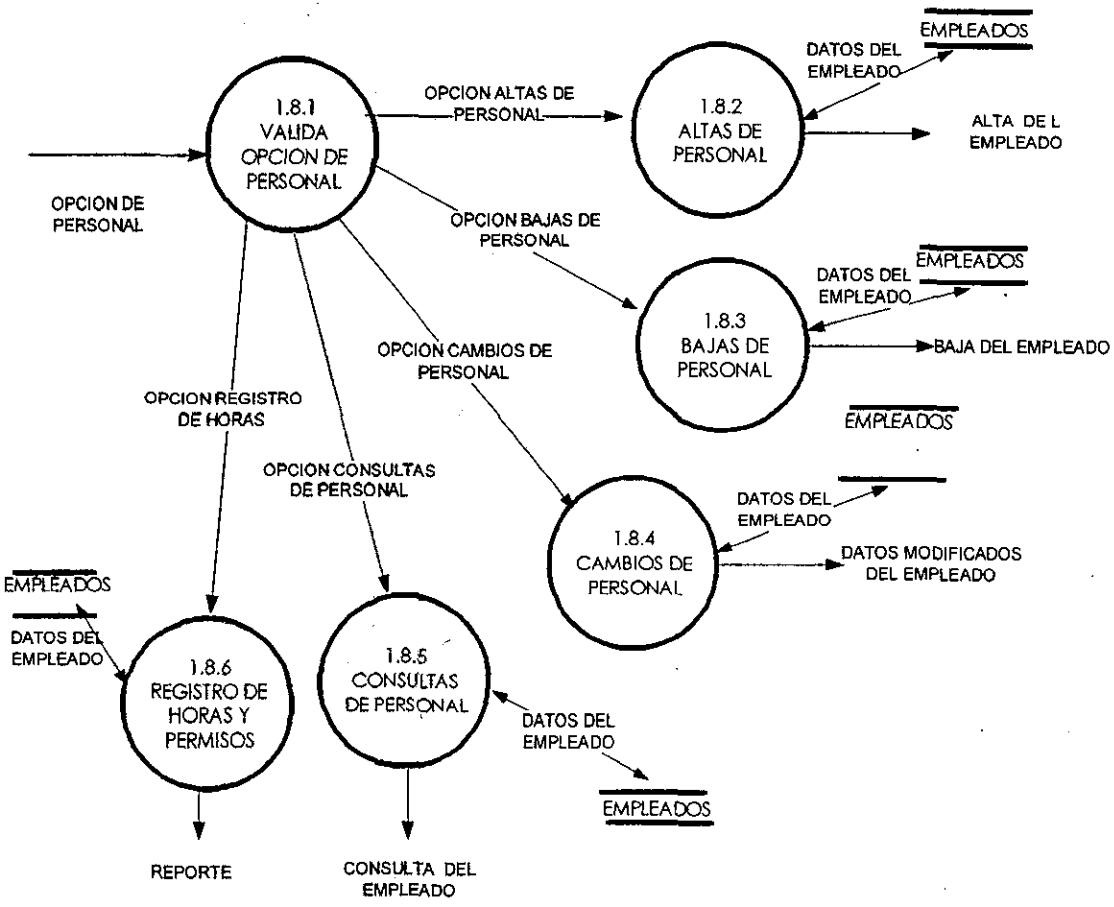



## MOVIMIENTO DE MATERIAL DE BIBLIOTECA
## NIVEL 1.6

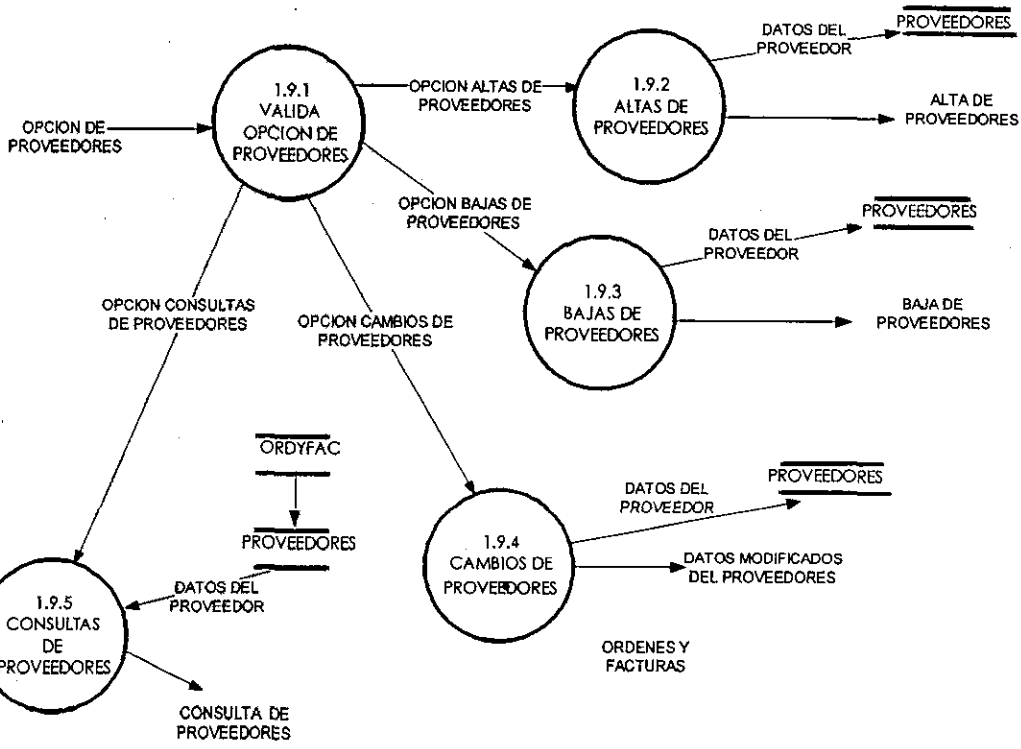


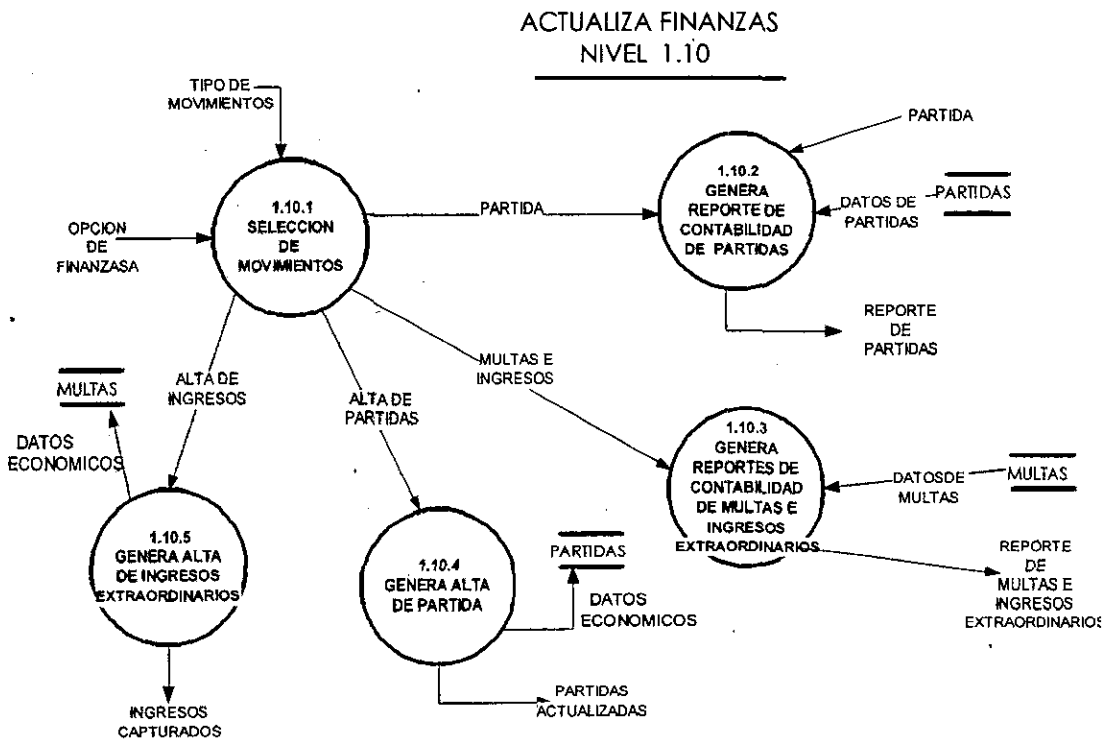

ACTUALIZA INVENTARIO
NIVEL 1.7
1.7.1
VALIDA
OPCION DE
INVENTARIO
OPCION DE
INVENTARIO
OPCION ALTAS DE
INVENTARIO
OPCION BAJAS DE
INVENTARIO
OPCION CAMBIOS DE
INVENTARIO
OPCION CONSULTAS
DE INVENTARIO
1.7.2
ALTAS DE
INVENTARIO
INVENTARIO
DATOS DE
INVENTARIO
ALTA DE
INVENTARIO
1.7.3
BAJAS DE
INVENTARIO
INVENTARIO
DATOS DE
INVENTARIO
BAJA DE
INVENTARIO
1.7.4
CAMBIOS DE
INVENTARIO
DATOS DE
INVENTARIO
INVENTARIO
INVENTARIO
MODIFICADO
1.7.5
CONSULTAS
DE
INVENTARIO
DATOS DE
INVENTARIO
CONSULTA
INVENTARIO

# ACTUALIZA PLANILLA DE PERSONAL
## NIVEL 1.8



## ACTUALIZA PROVEEDORES
## NIVEL 1.9






# DICCIONARIO DE DATOS

Se denomina diccionario de datos a las tablas que mantienen toda la relación de los campos y sus atributos asociados, es decir el nombre del campo, su longitud, tipo de dato, etc.

El proporcionar estándares a la base de datos, es un aspecto de la administración de la base de datos donde cada campo deberá tener un nombre, un formato y estrategia de acceso y cada archivo de la base de datos deberá tener relaciones con otros archivos estándar. La UNAM como organización cuenta con gran cantidad de estándares en cuanto a los datos que maneja, por lo que el sistema

debe contemplar que se respeten y se cumplan, tanto por nuevos sistemas o por modificaciones de uno ya existente, así que SABZ tiene que ser diseñado y analizado a partir de estos, y como a quedado establecido la DGB es la organización que coordina las bibliotecas y por ende es la que establece los estándares a seguir por aquellas que la conforma. Obviamente la DGB como elemento de la UNAM también debe apegarse a los estándares que rigen en la institución.

Una vez establecidos los detalles de la estandarización, son registrados en el Diccionario de datos.

**ALUMNOS**

| **Contenido** | **Tipo** | **Longitud** |
|---|---|---|
| Apellido Paterno | Alfanumérico | 25 |
| Apellido Materno | Alfanumérico | 25 |
| Nombre del Alumno | Alfanumérico | 25 |
| Clave de la Carrera | Numérico | 3 |
| Plan de estudios | Numérico | 2 |
| Sexo | Alfanumérico | 1 |
| | M = Masculino. | |
| | F = Femenino | |
| Cuenta | Alfanumerico | 9 |

**CARRERAS**

| **Contenido** | **Tipo** | **Longitud** |
|---|---|---|
| Clave de la Carrera | Numérico | 3 |
| Plan de estudios | Numérico | 2 |
| Nombre de la Carrera | Alfanumérico | 45 |

**PROFESORES**

| Contenido | Tipo | Longitud |
|---|---|---|
| Clave del Profesor | Alfanumérico | 13 |
| Nombre del Profesor | Carácter | 45 |

Conjuntando la información obtenida y el análisis de la misma, así como proceso de la base de datos que deberá estandarizase. Se obtiene el diccionario de datos del sistema SABZ.

**LECTORES** CATÁLOGO DE LECTORES

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| Apellido_P | vchar | 25 | Apellido Paterno | | |
| Apellido_M | vchar | 25 | Apellido Materno | | |
| Nombre | vchar | 25 | Nombre del lector | | |
| ↘✸ Carrera | sint | 3 | Clave de la carrera. | | I } |
| ↘✸ Plan | sint | 2 | Clave del plantel. | | I } |
| ✸ Cuenta | char | 9 | Número de cuenta. | | I } |
| Dir | vchar | 60 | Dirección del lector. | | |
| CP | char | 5 | Código postal. | | |
| Tel | vchar | 10 | Teléfono. | SI | |
| ↘ Tipolec | sint | 1 | Clave del tipo de lector. | | I |
| Fechafin | date | 8 | Fecha de vencimiento de credencial. | | |
| Material | sint | 2 | Cantidad de mat. prestado (libros, audiovisuales, etc.). | | |
| Extemp | sint | 2 | No. de libros entregados extemporaneamente. | | |
| Creden | char(1) | T/F | Indica status de credencial V = Inhibida, F = Des inhibida | | |
| Finhibi | date | 8 | Fecha de inhibición de credencial | | |
| Observa | char | 20 | Motivos de inhibición. | SI | |
| Multa | dec | 999.99 | Si tiene multa, y de cuanto. | | |
| Sem | int | 2 | Semestre actual del alumno | | |

**TIPOSLEC** CLASIFICACION DE LECTORES.

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Tipolec | sint | 1 | Tipo de lector. | | I |
| Limlib | sint | 2 | Limite de libros a prestar. | | |
| Limdías | sint | 2 | Limite de días de préstamo. | | |
| Limrefre | sint | 1 | Limite de resellos. | | |
| Descrip | vchar | 16 | Descripción del lector. | | |

**MULTAS** REGISTRA MONTO DE MULTAS.

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸Folio | char | 8 | # de folio de la multa. | | I |
| ✸Cuenta | char | 9 | Número de cuenta. | | I |
| Monto | dec | 999.99 | Monto de la multa. | | |
| Fecha | date | 8 | Fecha del pago de multa. | | |
| ➘ Status | sint | 2 | Procedencia del monto | | I |

**CARRERA** CATÁLOGO DE CARRERAS

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Carrera | sint | 3 | Clave de la carrera | | I |
| ✸ Plan | sint | 2 | Clave del plantel. | | I |
| Desc-Car | vchar | 26 | Descripción de la carrera. | | |

**FECVIG** FECHA DE LA VIGENCIA DE LA CREDENCIAL

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Fechafin | date | 8 | Vencimiento de creden. | | |

**FECINHA** DIAS INHABILES

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Fecinha | date | 8 | Fecha del día inhábil. | | |

**CATALOGO** DETALLE DEL MATERIAL BIBLIOGRAFICO

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Numadq | double | 7 | No. de adquisición. | | I |
| Matriz | double | 7 | No. de matriz del libro. | | |
| Clasific | vchar | 20 | Clasificación. | | I |
| ➘ Cveloc | sint | 1 | Clave de localización. | | |
| Título | vchar | 60 | Título del material. | | |
| Autor | vchar | 32 | Autor del material. | | |
| Coautor | vchar | 32 | Coautor del material. | SI | |
| ISBN | char | 10 | ISBN | | |
| Tema | vchar | 30 | Tema del libro. | SI | |
| Desfis | vchar | 10 | Descripción física (pag,Vol,etc.). | SI | |
| Nficha | double | 8 | No. ficha dado por DGB. | | |
| Idioma | vchar | 10 | Idioma del libro. | SI | |
| ➘ Status | sint | 2 | Status del materia (clave). | | I |
| Notas | vchar | 32 | Acerca del libro. | SI | |
| Fechaed | vchar | 5 | Fecha de edición. | SI | |
| ➘ Procedencia | sint | 1 | Status de procedencia | | I |

**LOCALI** LUGAR DE LOCALIZACION DEL MATERIAL

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cveloc | sint | 2 | Clave de localización. | | I |
| Locali | vchar | 20 | Lugar de localización. | | |
| Campo | char | 1 | Campo de la FESZ. | | |

**STATUS** STATUS DEL MATERIAL

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Status | sint | 2 | Status del material. | | I |
| Descrip | vchar | 20 | Descripción. | | |

**CONSTANCIA** FOLIO DE LA CONSTANCIA

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Folio | sint | 5 | Numero de Folio | | I |
| ➘ Cuenta | char | 9 | Cuenta del lector | | |
| ➘ Fecha | date | 8 | Fecha de expedición de constancia | | |
| ➘Tipo | char | 1 | Tipo de constancia | | I |

**MOVINV** MOVIMIENTOS REGISTRADOS DEL MATERIAL

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ➘ Status | sint | 2 | Status del material. | | I |
| Fecha | date | 8 | Fecha de vencimiento. | | |
| ➘✸Cveque | sint | 2 | Clave del tipo de material que se prestó. | | I } |
| ✸ Clave | vchar | 15 | Clave especifica de material | | I |
| Resello | sint | 1 | # de resellos del libro. | | |
| ➘✸Cuenta | char | 9 | No. de cuenta del lector. | | I } |
| ➘✸Carrera | sint | 3 | Clave de la carrera | | I |
| ➘✸Plan | sint | 2 | Clave del plan | | I } |

**QUE** QUE MATERIAL SE PRESTÓ

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cveque | sint | 2 | Clave del material que indica que se prestó. | | I |
| Que | vchar | 20 | Descripción del material. | | |

**ESTADISTICA** PERMITE LA ELABORACION DE ESTADISTICAS

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ➘✸Cveque | sint | 2 | Tipode material prestado | | I |
| ✸Clave | vchar | 15 | Clave especifica de material | | |
| Fecha | date | 8 | Fecha de movimiento. | | |

**USUARIOS** CATÁLOGO DE USUARIOS DEL SISTEMA

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cveusu | char | 8 | Clave de usuario. | | I |
| ✸ Cvecateg | sint | 2 | Número de usuario. | | |
| Categoria | vchar | 32 | Categoría del usuario. | | |

**MDIDAC** MATERIAL PRODUCIDO EN LA FESZ

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Nummat | char | 3 | No. del material. | | I |
| Tema | char | 20 | Tema básico del material. | | |
| Nivel | sint | 2 | Nivel | | |
| Sem | sint | 2 | Semestre | | |
| Modulo | sint | 2 | Modulo. | | |
| ➘ Carrera | sint | 3 | Clave de la carrera. | | I |
| ➘ Plan | sint | 2 | Plan de la carrera. | | I |
| Desfis | vchar | 10 | Descripción física | | |
| Notas | vchar | 32 | Acerca del libro. (pag,vol, etc). | SI | |
| Titulo | vchar | 32 | Titulo del material. | | |
| Autor | vchar | 32 | Autor. | | |
| Coautor | vchar | 32 | Coautor del material. | SI | |
| ➘ Cveloc | sint | 2 | Clave de localización. | | I |
| ➘ Status | sint | 2 | Status del materia (clave) | | I |
| Fecha | date | 4 | Fecha de elaboración | | |

**LTP** CATÁLOGO DE LABORATORIO Y TALLER DE PROYECTOS

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cveltp | char | 15 | Clasificación del LTP. | | I |
| Autor | vchar | 32 | Nombre del autor. | | |
| Titulo | vchar | 32 | Titulo del LTP. | | |
| Asesor | vchar | 32 | Nombre del asesor. | SI | |
| Año | vchar | 8 | Fecha de publicación. (mes/año(4)) | | |
| ➘ Status | sint | 2 | Status. | | I |
| ➘ Cveloc | sint | 2 | Clave de localización. | | I |

**AUDIOVISUALES** CATÁLOGO DEL MATERIAL AUDIOVISUAL

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cveaudio | char | 3 | Clave de audiovisual. | | I |
| Programa | vchar | 60 | Nombre del programa. | | |
| Sinte | vchar | 100 | Síntesis del programa. | SI | |
| Area | vchar | 15 | Area a la que pertenece. | | |
| Tema | vchar | 15 | Tema del programa. | | |
| Idioma | vchar | 10 | Idioma del programa. | SI | |
| Duración | sint | 3 | Minutos del programa. | | |
| Genero | vchar | 15 | Genero del programa. | SI | |
| ➘ Status | sint | 2 | Status del material. | | I |
| ➘ Cveloc | sint | 2 | Clave de localización. | | I |

**DIAPO** CATÁLOGO DE PROGRAMAS DE DIAPORAMAS

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cvedia | char | 3 | Clave de diaporama. | | I |
| Programa | vchar | 60 | Nombre del programa. | | |
| Diapo | sint | 3 | Número de diapositivas. | | |
| Apoyo | vchar | 15 | Tipo de apoyo que tiene. | SI | |
| Area | vchar | 15 | Area a la que pertenece. | | |
| ➘ Status | sint | 2 | Status del material. | | I |
| ➘ Cveloc | sint | 2 | Clave de localización. | | I |

**DIS-ESP** CATÁLOGO DE ESPACIOS.

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Idevento | int | 4 | Identificador del evento | | I |
| Evento | vchar | 32 | Nombre del evento | | |
| Horaini | time | | hora de inicio. | | |
| Horafin | time | | hora final. | | |
| Espacio | char | 2 | Sala a autorizar | | |
| Fecha | date | 8 | Fecha del evento | | |
| Personas | sint | 2 | Número de personas. | | |
| ➘ Cveque | sint | 2 | Especifica que material. | | I |
| ➘ Clave | vchar | 15 | Clave del material | | I |
| Nombre | vchar | 32 | Responsable. | | |

**MSSOCIAL** MATERIAL DE REPORTE DE SERVICIO SOCIAL

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cvess | char | 6 | Clave del material. | | I |
| Titulo | vchar | 32 | Titulo del reporte del Servicio Social. | | |
| Autor | vchar | 32 | Nombre del autor. | | |
| Institu | vchar | 32 | Institución. | | |
| Lugar | vchar | 32 | Lugar de la institución. | SI | |
| ➘ Status | sint | 2 | Status del material. | | I |
| ➘ Cveloc | sint | 2 | Clave de localización. | | I |
| ➘ Carrera | sint | 3 | Clave de la carrera. | | I |
| ➘ Plan | sint | 2 | Clave de plan. | | I |

**REVISTAS** CATÁLOGO DE REVISTAS

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cverev | char | | Clave de identificación. | | I |
| Titulo | vchar | 32 | Titulo de la revista. | | |
| Frecuencia | vchar | 5 | Cada cuando se compra. | | |
| Editor | vchar | 20 | Editor | | |
| Dir | vchar | 60 | Dirección. | | |
| Instituto | vchar | 20 | Instituto encuadernación | | |
| Fecha | char | 4 | Año de publicación. | | |
| Seriacion | int | | Serie de la revista. | | |
| Volumen | int | | Volumen de la revista. | SI | |
| Cantidad | int | 2 | Cantidad por mes. | | |
| Observaciones | vchar | 32 | Observaciones generales. | SI | |
| ➘ Status | sint | 2 | Status del material. | | I |
| ➘ Cveloc | sint | 2 | Clave de localización. | | I |
| ➘ Procedencia | sint | 1 | Status de procedencia. | | I |
| Numero | double | 6 | Identificación de revista. | | |
| Pais | vchar | 15 | País de origen. | | |
| ISNN | char | 8 | Identificación Internacional | | |
| Idioma | vchar | 15 | Idioma de la revista | | |

**TESIS** CATÁLOGO DE TESIS.

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cvetes | char | 15 | Clasificación de la tesis. | | I |
| Titulo | vchar | 32 | Titulo de la tesis. | | I |
| Autor | vchar | 60 | Nombre de/o autore(s). | | I |
| Asesor | vchar | 32 | Nombre del asesor. | | |
| Año | date | 8 | Año que se realizo. | | |
| Lugar | vchar | 15 | Lugar donde se realizo. | | |
| Noejem | sint | 1 | Número de ejemplares. | | |
| ➘ Status | sint | 2 | Status de la tesis. | | I |
| ➘ Cveloc | sint | 2 | Clave de localización. | | I |
| ➘ Carrera | sint | 3 | Clave de la carrera. | | I |
| ➘ Plan | sint | 2 | Plan de la carrera. | | I |

**MAPOTECA** CATÁLOGO DE MAPOTECA.

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cvemap | char | 7 | Clave de INEGI. | | I |
| Nombre | vchar | 20 | Nombre de la región. | | |
| Estado | vchar | 20 | Nombre del estado. | | |
| Upganado | char(1) | T/F | Si hay mapa. | | |
| Upfores | char(1) | T/F | Si hay mapa. | | |
| Upagric | char(1) | T/F | Si hay mapa. | | |
| Geologia | char(1) | T/F | Si hay mapa. | | |
| Uso-suelo | char(1) | T/F | Si hay mapa. | | |
| Edafologia | char(1) | T/F | Si hay mapa. | | |
| Ha_superf | char(1) | T\F | Si hay mapa. | | |
| Ha_subter | char(1) | T\F | Si hay mapa. | | |
| Topografia | char(1) | T\F | Si hay mapa. | | |
| C-may-oct | char(1) | T\F | Si hay mapa. | | |
| C-nov-abr | char(1) | T/F | Si hay mapa. | | |
| Escala | vchar | 8 | Escala del mapa. | | |
| Loca_fis | vchar | 2 | Ubicación del mapa. | | |
| ➘ Cveloc | sint | 2 | Clave de localización | | I |
| ➘ Status | sint | 2 | Status del material | | I |

**PROCMAT** STATUS DE PROCEDENCIA DEL MATERIAL.

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Procedencia | sint | 1 | Status de procedencia | . | I |
| Descripcion | vchar | 32 | Descripción. | | |

**EMPLEADOS** CATÁLOGO DE EMPLEADOS

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cuenta | char | 9 | Número de trabajador. | | I |
| RFC | char | 13 | RFC del empleado. | | |
| Empleado | vchar | 32 | Nombre del empleado. | | |
| Sexo | char(1) | M/F | Sexo. | | |
| Dir | vchar | 60 | Dirección del trabajador. | | |
| Tel | vchar | 10 | Teléfono del empleado. | SI | |
| CP | char | 5 | Código postal. | | |
| Salario | Dec | 8.2 | Salario del empleado. | | |
| Horario | vchar | 14 | Horario de servicio. | | |
| Añoin | date | 8 | Fecha de contratación. | | |
| ↘ Cvecateg | sint | 2 | Clave de categoría. | | I |
| ↘ Cvedep | sint | 2 | Clave del departamento. | | I |
| Permisos | sint | 2 | # de días económicos. | | |

**DEPTOS** CATÁLOGO DE DEPARTAMENTOS

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cvedep | sint | 2 | Clave de departamento. | | I |
| Depto | char | 32 | Nombre del depto. | | |

**HORAS** DE HORAS TRABAJADAS POR EMPLEADO

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| Numhor | sint | 3 | # de horas trabajadas. | | |
| Numext | sint | 3 | Número de horas extras. | | |
| Semana | date | 8 | Semana actual. | | |
| ✸ RFC | char | 13 | RFC del empleado. | | I |

**PERMISOS** PERMISOS DE PERSONAL.

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ RFC | char | 13 | RFC del empleado. | | I |
| Fecha | date | 8 | Fecha del permiso | | |

**CATINV** CATÁLOGO DE INVENTARIO.

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸Cveart | char | 10 | Clave del articulo. | | I |
| Descrip | vchar | 30 | Descripción. | | |
| Existencia | double | 8 | Numero de existencia. | | |

**INVENTARIO** DETALLE DEL INVENTARIO.

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸Cveart | char | 10 | Clave del articulo. | | I |
| Desdet | vchar | 30 | Detallado del articulo. | | |
| Inventario | double | 8 | Número de inventario. | | |
| Esp_fis | vchar | 9 | Espacio físico. | | |
| Cantidad | sint | 4 | Cuantos hay. | | |

**PROVEEDORES** CATÁLOGO DE PROVEEDORES.

| NOMBRE | TIPO | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|
| RFC | char 10 | RFC del proveedor. | | |
| ✸Cvepro | char 15 | clave del proveedor. | | I |
| Nombre | vchar 32 | Nombre del proveedor. | | |
| RazónS | vchar 50 | Razón social de la empresa | | |
| Dir | vchar 60 | Dirección del proveedor. | | |
| Tel | char Ext[2]10 | Teléfono directo. | | |
| Fax | char 10 | Fax del proveedor | | |
| CP | char 5 | Código postal. | | |
| Email | char 20 | Correo electrónico | SI | |
| Añocon | date 8 | Fecha de contratación. | | |
| Representante | vchar 32 | Nombre Representante. | | |
| Horario. | char 8 | Horario de servicio. | SI | |
| Tipo_des | dec | Porcentaje de descuento. | | |
| Exclusivo | char(1) T/F | Bandera de exclusividad. | | |
| Nacionalidad | vchar 15 | Sea nacional o extranjero | | |
| Tiempo_en | vchar 15 | Tiempo de entrega del material. | | |

**ORDYFAC** ORDENES DE COMPRA Y FACTURA.

| NOMBRE | TIPO | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|
| ✸Bandera | sint 2 | Bandera factura/pedido | | I |
| ✸➘ Cveorfac | vchar 15 | Clave de orden o factura | | I |
| ➘ Status | sint 2 | Status del material > 20. | | I |
| ➘ Cvepro | char 15 | Clave del proveedor | | I |
| Fec_ped | date 8 | Fecha de pedido. | | |
| Fec_ènt | date 8 | Fecha de entrega. | | |
| Total | dec 8.2 | Monto total de la orden. | | |
| ➘ Partida | char 15 | Código de la partida. | | I |
| Total_tit | sint 2 | Total ejemplares pedidos. | | |
| Moneda | vchar 15 | Nombre de la moneda | | |
| Tipo_cam | dec 5.2 | Tipo de cambio | | |
| Adelanto | dec 6.2 | Adelanto monetario. | | |
| Impreal | dec 8.2 | Cantidad surtida real (si fue parcialmente surtida) | | |
| Campo | char 1 | Campo de la facultad. | | |
| Marca | char(1) T/F | Indica asociada NC | | I |

**DETALLE** DETALLE DE ORDENES Y FACTURAS.

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸Cveorfac | vchar | 15 | Clave de la factura. | | I |
| Titulo | vchar | 60 | Titulo solicitado. | | |
| ISBN | char | 10 | ISBN. | | |
| Autor | vchar | 60 | Autor del libro. | | |
| Coautor | vchar | 60 | Coautor del libro. | SI | |
| Editorial | vchar | 15 | Editorial de la obra. | | |
| Edicion | sint | 2 | # de edición de la obra. | | |
| Fec_pub | date | 8 | Año de publicación. | | |
| Num_ejem | sint | 3 | # ejemplares del titulo. | | |
| Num_vol | sint | 2 | # de volúmenes por tit. | | |
| Costo | dec | 6.2 | Costo por ejemplar. | | |
| ↘ Carrera | sint | 3 | Clave de la carrera. | | I |
| ↘ Plan | sint | 2 | plan de la carrera | | I |
| ✸ NS | sint | 3 | Número de solicitud. | | |
| Idioma | vchar | 15 | Idioma de la obra. | | |
| ↘ Status | sint | 2 | Status de la solicitud. | | I |
| Lugar | vchar | 60 | Lugar de publicación. | | |
| Ejem_surtidos | sint | 3 | # de ejemplares surtidos. | | |

**BONO** DETALLADO DE LA NOTA DE CREDITO.

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cveorfac | vchar | 15 | Factura que lo origina. | | I |
| Oryfac | vchar | 15 | Factura que lo utiliza. | | |
| Bono | char | 15 | Número de bono. | | |
| Cantidad | dec | 6.2 | Cantidad de la nota | | |
| Fecha | date | 8 | Fecha de expedición. | | |
| Concepto | vchar | 30 | Observación. | | |
| Aplicada | char(1) | T/F | Bandera de aplicación si existe nota de crédito. | | |
| Tipo_cam | dec | 6.2 | Tipo de cambio. | | |
| Moneda | vchar | 15 | Nombre de la moneda. | | |

**OFMYE** ORDENES Y FACTURA DE MOBILIARIO Y EQUIPO.

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Bandera | char | 1 | Bandera factura/pedido | | I |
| ✸↘ Cveorfac | vchar | 15 | Clave de orden y factura | | I |
| ↘ Status | sint | 2 | Status del material > 40. | | I |
| Fec_ped | date | 8 | Fecha de pedido. | | |
| Fec_ent | date | 8 | Fecha de entrega. | | |
| ↘ Partida | char | 15 | Código de la partida. ser parcialmente surtida) | | I |
| Campo | char | 1 | Campo de la facultad. | | |
| Total | dec | 8.2 | Total de la compra. | | |

**DETMYE** DETALLADO DE FACTURAS DE MOBILIARIO Y EQUIPO

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Cveorfac | vchar | 15 | Clave de la factura. | | I |
| Descrip | vchar | 60 | Descripción del material | | |
| Cantidad | dec | 7.2 | Costo total del pedido. | | |
| Observaciones | vchar | 60 | Nota del pedido. | | |
| Justif | vchar | 60 | Justificación el pedido. | | |
| Nodoc | sint | | Documento (folio) | | |
| Campo | char | 1 | Campo de la facultad. | | |
| ✸ Nsoli | sint | 2 | Numero de solicitud | | |

**PARTIDA** DESCRIPCION DE CONTABILIDAD DE PARTIDAS

| NOMBRE | TIPO | | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|---|
| ✸ Partida | char | 15 | Código de la partida. | | I |
| Cantidad | dec | 8.2 | Cantidad real destinada. | | |
| Comprometida | dec | 8.2 | Cantidad comprometida. | | |
| Ejercida | dec | 8.2 | Cantidad ejercida. | | |
| Fecha | date | 8 | Fecha de actualización | | |
| Tbonos_ap | sint | 3 | Bonos aplicados. | | |
| Tbonos_d | sint | 3 | Bonos disponibles. | | |

**RECIBO** CONTRA – RECIBO DE FACTURAS.

| NOMBRE | TIPO | DESCRIPCIÓN | NULL | INDICE |
|---|---|---|---|---|
| ✱Id_recibo | sint | Identificador del contra - recibo. I | | I |
| cveorfac | vchar 15 | Factura. que lo origina | | |

Indice compuesto por dos columnas.

Indice compuesto por tres columnas.

✱ Llave primaria.
➘ Llave foránea.

## DESCRIPCION DE ARCHIVOS.

**CARRERA** CATÁLOGO DE LAS CARRERAS.

| Carrera | Plantel | Desc-carr |
|---|---|---|
| 503 | 22 | Biólogo. |
| 505 | 21 | Ingeniero Químico. |
| 505 | 24 | Q. F. B. |
| 512 | 08 | Medico Cirujano. |
| 509 | 01 | Enfermería. |
| *** | ** | Lic.Enfermería |
| 514 | 21 | Odontología. |
| 519 | 21 | Psicología. |
| 001 | 01 | Maestría. |
| 002 | 02 | Diplomado. |
| 003 | 03 | Doctorado. |
| 004 | 04 | Seminario. |
| 005 | 05 | Exalumno. |
| 006 | 06 | Académico. |
| 007 | 07 | Administrativo. |
| 008 | 08 | Funcionario. |
| 009 | 09 | Biblioteca. |
| 010 | 10 | Investigador |

**TIPOSLEC** CLASIFICACION DE LECTORES

| Tipolec | Limlib | Limdias | Limrefre | Descrip |
|---|---|---|---|---|
| 1 | 4 | 5 | 1 | Alumno |
| 2 | 4 | 10 | 1 | Académico |
| 3 | 4 | 5 | 1 | Administrativo |
| 4 | 4 | 10 | 1 | Investigador |
| 5 | 4 | 5 | 1 | Funcionarios |
| 6 | 4 | 5 | 1 | Posgrado |
| 7 | 4 | 5 | 1 | Biblioteca |

Nota : El ultimo día laborable, antes del periodo de vacaciones, se prestan 6 libros.

**LOCALI** CATÁLOGO DE LOCALIZACION.

| Cveloc | Locali | Campo I, II |
|---|---|---|
| 0 | Acervo | I |
| 1 | Sala de Consulta | I |
| 2 | Hemeroteca | I |
| 3 | Audiovisuales | I |
| 4 | Procesos Técnicos | I |
| 10 | Acervo | II |
| 11 | Sala de Consulta | II |
| 12 | Hemeroteca | II |
| 13 | Audiovisuales | II |
| 14 | Procesos Técnicos | II |
| 15 | Mapoteca | II |

**PROCMAT** STATUS DE PROCEDENCIA DEL MATERIAL

| Procedencia | Descripción |
|---|---|
| 0 | Compra |
| 1 | Donación |
| 2 | Suscripción |
| Procedencia de dinero | |
| 30 | Reposición de credencial |
| 31 | Multa por atraso |
| 32 | Inscripción |

**STATUS** STATUS DEL MATERIAL.

| | Status | Descripción. |
|---|---|---|
| LIBROS. | | |
| | 0 | Encuadernación. |
| | 1 | Prestado. |
| | 2 | Procesos Técnicos. |
| | 3 | Mutilado. |
| | 4 | Robo. |
| | 5 | Perdido. |
| | 6 | Descartado. |
| | 7 | Litigio. |
| | 8 | Baja. |
| | 9 | Reposición Modificada |
| | 10 | Acervo. |
| | 11 | Solo Consulta. |
| PEDIDOS | | |
| | 20 | Ingresado. |
| | 21 | Pedidos. |
| | 22 | Surtido. |
| | 23 | Parcialmente Surtido. |
| | 24 | Cancelado. |
| | 25 | Reordenado. |

**QUE** QUE MATERIAL SE PRESTO.

| Cveque | Que |
|---|---|
| 0 | Libro. |
| 1 | Revista. |
| 2 | Audiovisuales |
| 3 | Diaporamas. |
| 4 | Tesis. |
| 5 | Servicio Social. |
| 6 | LT.P |
| 7 | Mapas. |
| 8 | Material Didáctico. |
| 9 | Equipo |

# BASE DE DATOS

| ARCHIVOS | DESCRIPCIÓN |
|---|---|
| **USUARIOS** | Catálogo de usuarios del sistema. |
| **TIPOSLEC** | Clasificación de lectores. |
| **LECTORES** | Catálogo de lectores. |
| **CONSTANCIA** | Folio de constancias generadas. |
| **CARRERA** | Catálogo de carreras. |
| **MOVINV** | Movimientos que el sistema registra del material. |
| **ESTADISTICA** | Movimientos que el sistema registra para la elaboración de las estadísticas del material bibliográfico. |
| **MULTAS** | Registra el monto y el folio correspondiente de las multas. |
| **LOCALI** | Lugar de localización del material. |
| **STATUS** | Status del material. |
| **FECVIG** | Fecha de la vigencia de la credencial. |
| **FECINHA** | Contiene las fechas de los días inhábiles. (incluyendo domingos) Actualizado cada semestre o año. |
| **QUE** | Descripción del material. |
| **PROCMAT** | Status de procedencia del material. |
| **CATÁLOGO** | Detalle del material bibliográfico. |
| **MDIDAC** | Catálogo del material producido en la FES Zaragoza. |
| **AUDIOVISUAL** | Catálogo del material de audiovisual existente |
| **DIAPO** | Catálogo del material de diaporamas. |
| **REVISTAS** | Catálogo de revistas. |
| **MAPOTECA** | Catálogo de mapas. |
| **MSSOCIAL** | Catálogo de material de servicio social. |
| **TESIS** | Catálogo de tesis. |
| **LTP** | Catálogo de LTP (Laboratorio y Taller de Proyectos) |
| **DIS-ESP** | Distribución de espacios audiovisuales (salas). |
| **CATINV** | Catálogo de inventario. |
| **INVENTARIO** | Detallado de inventario. |
| **EMPLEADOS** | Catálogo de empleados de biblioteca. |
| **DEPTOS** | Descripción de departamentos internos de biblioteca. |
| **HORAS** | Detallado de horas extras trabajadas por empleado. |
| **PERMISOS** | Registra fecha de permisos a empleados. |
| **PROVEEDORES** | Catálogo de proveedores. |
| **ORDYFAC** | Registro de las ordenes y números de facturas de mat. bibliográfico |
| **DETALLE** | Detallado de ordenes y facturas del Mat. Bibliográfico. |

**continuación**

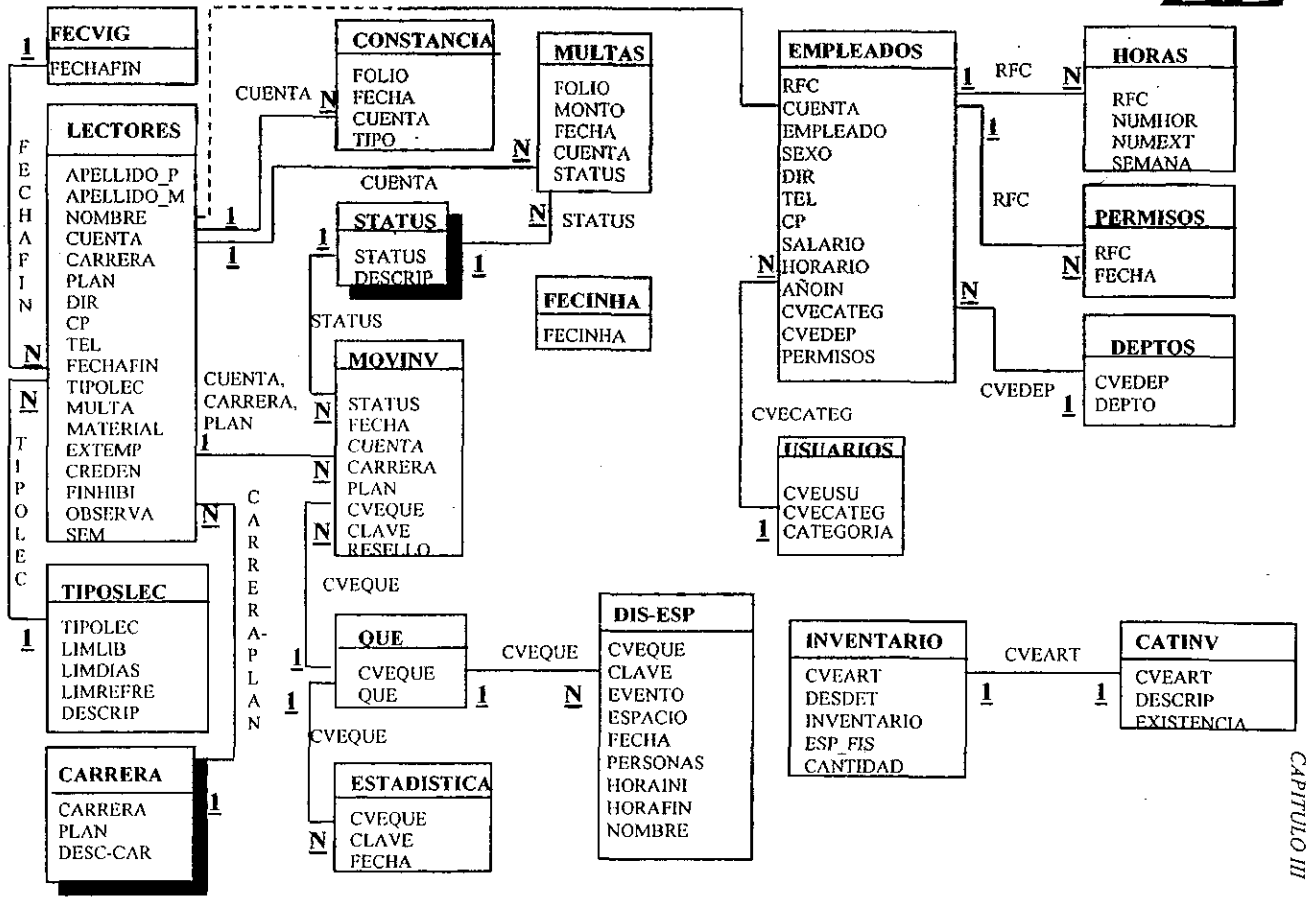
| ARCHIVOS | DESCRIPCIÓN |
|---|---|
| **OFMYE** | Registro de las ordenes y numero de facturas de material y equipo. |
| **DETMYE** | Detallado de ordenes y facturas del material y equipo. |
| **BONO** | Datos de las notas de crédito. |
| **PARTIDA** | Descripción de movimientos de partidas. |
| **RECIBO** | Contra – Recibo de las facturas. |

## DIAGRAMA ENTIDAD - RELACIÓN

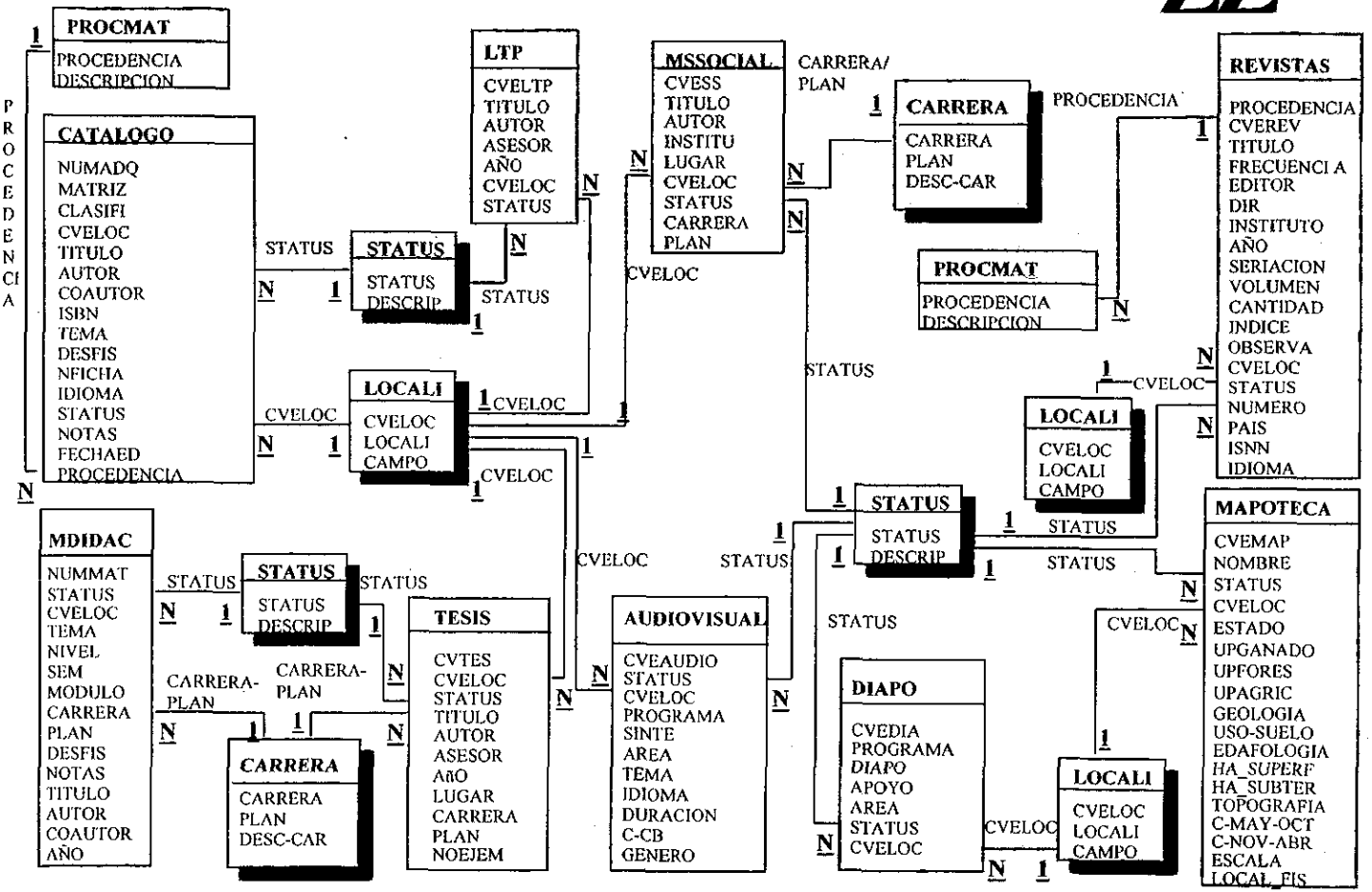
El diagrama Entidad-Relación se desarrolla a partir del análisis de requerimientos, surgiendo así reglas y validaciones. Con este modelo se definen las entidades, los atributos y las relaciones, las cuales son asociaciones entre entidades. (Las relaciones pueden ser de diferentes grados, dependiendo de las entidades que participan en ella, la mayoría son binarias).

A continuación se define el diagrama E-R de SABZ.

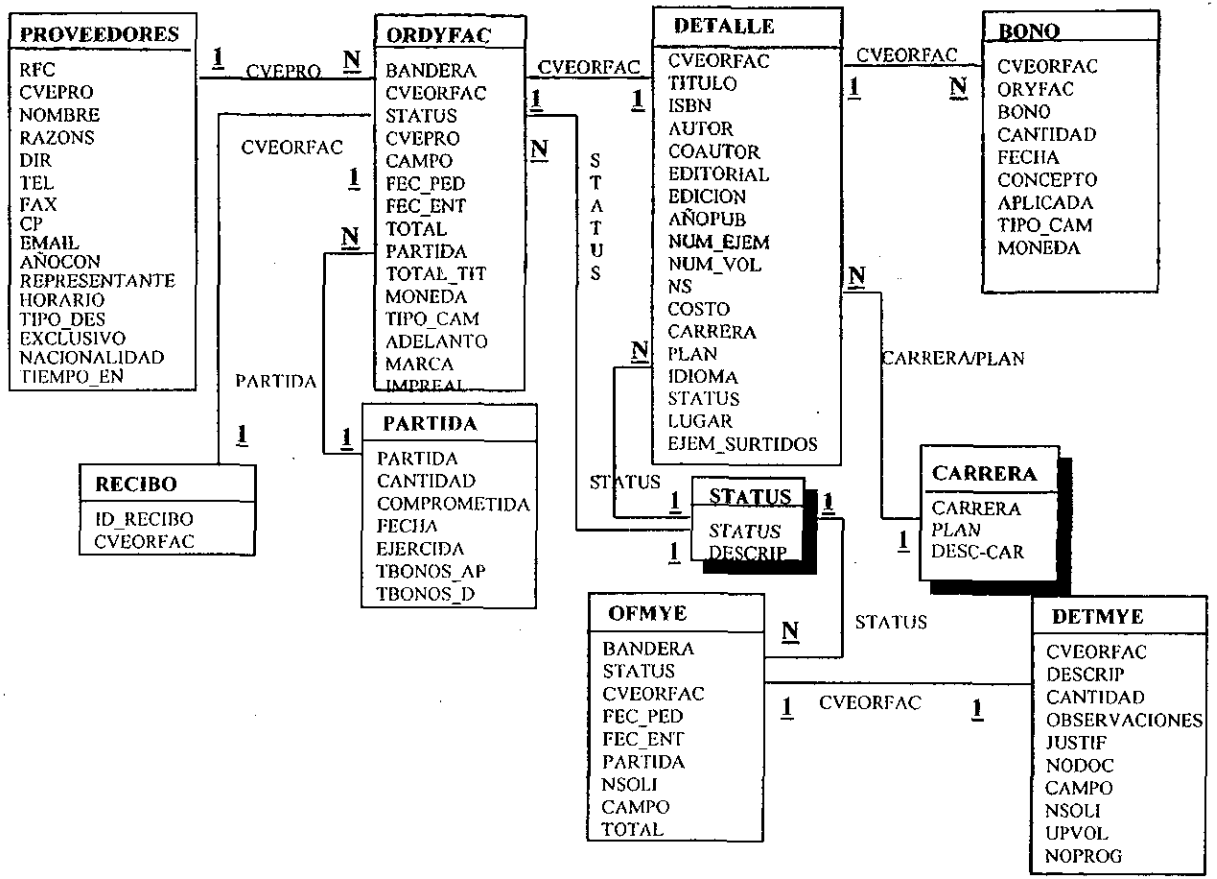
# SISTEMA AUTOMATIZADO DE BIBLIOTECAS "SABZ"
## FACULTAD DE ESTUDIOS SUPERIORES "ZARAGOZA"
## DIAGRAMA ENTIDAD-RELACION



# SISTEMA AUTOMATIZADO DE BIBLIOTECAS "SABZ"
# FACULTAD DE ESTUDIOS SUPERIORES "ZARAGOZA"
## DIAGRAMA ENTIDAD-RELACION



# SISTEMA AUTOMATIZADO DE BIBLIOTECAS "SABZ"
# FACULTAD DE ESTUDIOS SUPERIORES "ZARAGOZA"
## DIAGRAMA ENTIDAD-RELACION



# III.3 CODIFICACIÓN.

La programación o codificación consiste en tomar la documentación elaborada para el sistema durante el diseño y traducirlos a programas que se procesen y ejecuten en la computadora.

La codificación es una consecuencia natural del diseño. Sin embargo, las características del lenguaje y estilo de programación pueden afectar profundamente la calidad y el mantenimiento del sistema.

Como la herramienta que se utilizará para la interface gráfica de usuario (GUI) es PowerBuilder v.5, que cuenta con un lenguaje propio de programación (denominado Power Script) se debe sujetar la codificación del sistema a la sintaxis que dictamina este lenguaje. Independientemente de las sentencias SQL embebidas dentro de los programas para poder hacer uso de los archivos de las bases de datos por medio del DBMS.

Esta etapa se lleva a cabo a partir del diseño detallado del funcionamiento del sistema, pero descansa indudablemente en la experiencia e ingenio para con la elaboración de los algoritmos pertinentes que producirán las salidas requeridas, así como aceptar si y sólo si aquellas entradas que cumplan con las reglas de validación, creadas anticipadamente, previniendo falta de concordancia en los datos.

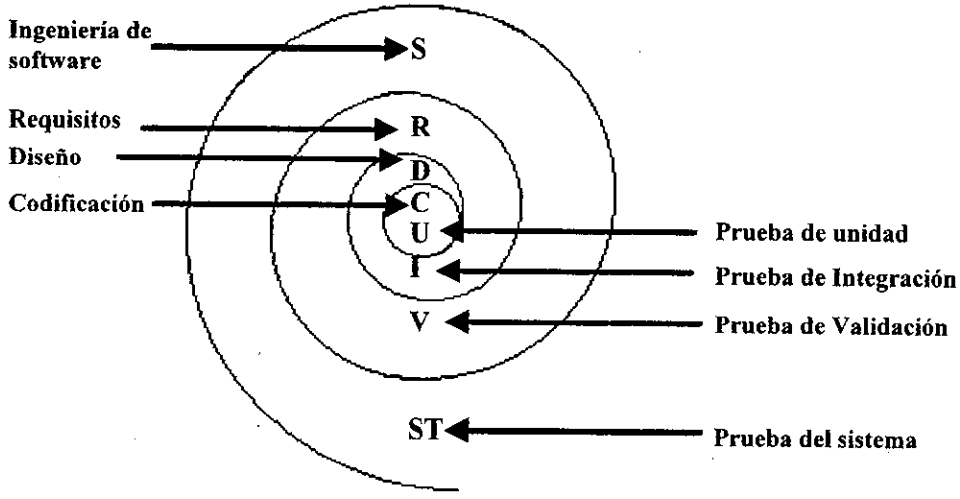
La complejidad y la organización de las estructuras de datos se definen durante el paso de diseño. El estilo en la declaración de los datos se establece cuando se genera el código.

## III.4 PRUEBAS.

La prueba del sistema es un elemento crítico que garantiza de manera esencial la calidad del software y representa una revisión final de las especificaciones de diseño y de la codificación. La prueba es un conjunto de actividades que se pueden planificar por adelantado y llevar a cabo sistemáticamente.

El proceso de desarrollo de sistema se puede ver como una espiral, donde inicialmente se define el papel del software y conduce al análisis de los requisitos del sistema, donde se establece el campo de información, funciones, rendimiento, restricciones, criterios de validación, etc. Enseguida se tiene el diseño y por último la codificación. Al desarrollar un sistema, se mueve hacia adentro de la espiral donde va disminuyendo en nivel de abstracción.

Ahora, una estrategia de prueba del sistema sería el movernos en sentido inverso, esto es partir del vértice de la espiral y centrarse en cada ***unidad*** del sistema, tal como esta implantado el código fuente. En la prueba de ***integración*** el foco de atención es el diseño y las relaciones entre los componentes. Al movernos por la espiral hacía afuera, se tiene la prueba de ***validación***, donde se validan los requisitos establecidos como parte del análisis de los requisitos del sistema comparándolos con el sistema construido. Y finalmente se llega a la prueba del ***sistema*** en la que se prueba como un todo el sistema y otros elementos que interactuaran con el. Para probar el software se mueve hacía afuera de la espiral, que a cada vuelta aumenta el alcance la prueba.




- **Prueba de Unidad.** Centra el proceso de validación en la menor unidad del diseño – el módulo- usando la descripción del diseño como guía, se prueban los caminos del control con el fin de cubrir errores dentro del ámbito del módulo. Los módulos se prueban independientes uno del otro permitiendo detectar errores en la codificación y la lógica que lo conforman.

- **Prueba de Integración.** Se centra en la incorporación de los módulos en la estructura de enlace y comunicación, permitiendo que cada uno de los módulos trabajen de forma relacional con el resto de ellos. Se comprueba:

  *La integridad de interfaz.* Se prueban las interfaces internas y externas a medida que se incorpora cada módulo a la estructura.
  *Validez funcional.* Se llevan a cabo pruebas diseñadas para descubrir errores funcionales.
  *Contenido de la información.* Se hacen pruebas diseñadas para descubrir errores asociados con las estructuras de datos globales o locales.
  *Rendimiento.* Se realizan pruebas con el fin de verificar los. limites de rendimiento establecidos durante el diseño del software.

- **Prueba de Validación**. Demuestra el seguimiento de los requerimientos del sistema. Cuando se construye software a la medida se deben de llevar a cabo una serie de pruebas de aceptación (Básicamente se realizan las pruebas alfa y beta) donde el usuario final valida los requisitos y descubre errores que parece que solo él puede descubrir.

  *Prueba Alfa.* Esta es realizada por el usuario final en el lugar del desarrollo, donde el equipo de desarrollo registra los errores y problemas de uso del sistema (Las pruebas alfa se realizan en un entorno controlado).
  *La prueba Beta.* El sistema se distribuye a varios usuarios, quienes lo prueban bajo condiciones reales, es decir, el sistema esta en vivo, excepto que los usuarios saben que están utilizando un sistema que puede fallar. El usuario registra todos los problemas que encuentra y da un informe a intervalos regulares.

- **Prueba del Sistema**. Valida el software una vez que se ha incorporado a un sistema mayor a través de una serie de pruebas, cuyo propósito principal es ejercitarlo profundamente, verificando que se han integrado adecuadamente los elementos y que realizan las funciones apropiadas.

  *Prueba de recuperación.* Mediante la prueba de recuperación se fuerza al sistema a fallar por medio de diversas formas y se verifica que la recuperación se lleve acabo apropiadamente.
  *Prueba de seguridad.* La prueba de seguridad intenta verificar que los mecanismos de protección incorporados en el sistema lo protegerán de cualquier tipo de penetración impropia.
  *Prueba de resistencia.* Esta prueba ejecuta el sistema de forma que demande recursos en cantidad, frecuencia en volúmenes anormales, se pretende descubrir

combinaciones de datos de entrada válida que puedan producir inestabilidad o procesamiento incorrecto.

*Prueba de rendimiento.* Diseñada para probar el rendimiento del sistema en tiempo de ejecución dentro de un contexto integrado.

La retroalimentación de esta fase de prueba produce cambios en el sistema para manejar los errores y fallas que se descubren.

Finalmente se crea un conjunto de datos que el sistema procesará como una entrada normal, sin embargo los datos son creados con la intención expresa de determinar si el sistema los procesará correctamente. Los datos de prueba cubren todo un espectro de situaciones de proceso. Primero se procesan datos de prueba típicos para ver si el sistema puede trabajar situaciones normales, luego se agregaran variaciones que incluirán datos inválidos para asegurar que el sistema puede detectar los errores de manera adecuada.

Cuando la prueba del sistema con "**datos de prueba**" llega a ser satisfactoria, es conveniente hacer que el sistema opere con "**datos reales**", permitiéndonos tener una visión real del funcionamiento del sistema.

Sin lugar a dudas ***"la prueba nunca termina, simplemente se transfiere del encargado del sistema al cliente"***. Toda vez que el cliente usa el programa, lleva a cabo una prueba.

La documentación proporciona un panorama del sistema, los procedimientos que debe realizar para operarlo, y detalla el código de los programas utilizados, esto es porque la documentación correspondiente de un sistema presenta una descripción del diseño de este. **"La documentación consiste en el material que explica las características y operación del sistema."**

El diseño es técnicamente la parte central de la ingeniería de software. Durante esta etapa se desarrollan, se revisan y se documentan los progresivos refinamientos de las estructuras de datos, de la estructura del programa y de los detalles procedimentales.

La fase de diseño se concluye con un entendimiento bastante claro de lo que se desea y de lo que se tendrá que hacer para obtenerlo. Así que el diseño completa toda la información requerida para la implantación total de la estructura de datos y su contenido.

Enfatizando, el análisis y diseño de sistemas, es el proceso que consiste en estudiar una situación a través de una recopilación de información, que forma la base para crear las estrategias alternativas de diseño y de solución.

# CAPITULO IV

## IMPLEMENTACION

Finalmente, el sistema debe ponerse en condiciones de trabajar. El hecho de tener todos los componentes juntos, no significa que trabajarán juntos sin altibajos o que los usuarios se adaptarán a la nueva manera de hacer las cosas de inmediato. El desarrollo de nuevos programas, convirtiendo los viejos recursos en nuevos y capacitando al personal para utilizar el nuevo sistema, son todos parte de la fase final del desarrollo, la implantación[1].

Implantación. Esta fase incluye todas las tareas restantes que son necesarias para ser operante y exitoso al sistema incluyendo programación y conversión de archivos, depuración y documentación, capacitación, conversión del sistema, retroalimantación, evaluación y mantenimiento.

El repaso de las metas y el alcance que se consideró en la fase de análisis se realiza en esta etapa y es importante puesto que las metas y el alcance definen respectivamente la dirección y los limites.

[1] **Implementación** no es una palabra reconocida en el lenguaje castellano, sin embargo se maneja como sinónimo de implantación y dado que es de uso común en el ámbito computacional adoptar la palabra implementación, el presente capitulo así se denomina.

# IV.2 DEFINICION DE LA ESTRUCTURA DE LA BASE DE DATOS PARA EL DBMS.

Existen varios métodos distintos mediante los cuales la estructura de una base de datos se describe para el DBMS, pero depende del DBMS que se utilice. En algunos casos se crea un archivo de texto, que describe la estructura de la base de datos, el lenguaje utilizado para escribir dicha estructura se conoce como lenguaje de definición de datos, es decir DDL (Data Definition Language), el archivo de texto DDL da los nombres de las tablas, describe las columnas, define los índices y describe también otras estructuras como limitantes y restricciones de seguridad. Algunos productos DBMS no requieren del archivo DDL en formato de archivo de texto, una alternativa común es que proporcionan un medio gráfico para la definición de la estructura de la base de datos

Pero, independientemente del medio a través del cual sea definida la estructura de la base de datos, se deberá identificar cada tabla, definir las columnas, describir el formato físico para cada columna, además dependiendo de las facilidades del DBMS, se podrán especificar limitantes que el DBMS deberá cumplir, por ejemplo: los valores de las columnas pueden quedar definidos como NOT NULL (NO NULO) o UNIQUE (UNICO), etc.

## SYBASE. SQL SERVER

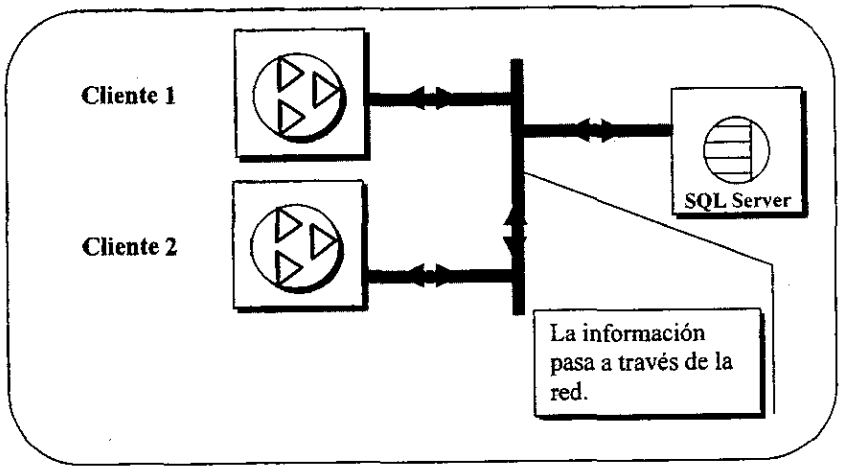
Sybase system 10 ofrece una variedad de características importantes para los administradores de sistema, las más notables son una capacidad más rápida para realizar la copia de seguridad y auditorías. SQL Server corre bajo varias

plataformas operativas incluyendo UNIX, NetWare, Windows NT, OS/2, Open VMS. UNIX es por mucho la plataforma dominante para Sybase SQL Server, y los equipos más apropiados son por tanto Sun, HP, IBM, DEC y AT&T.

SQL Server Características/Funciones:

- Los clientes de SQL Server pueden correr en cualquier plataforma del servidor.
- Administra conexiones de usuarios.
- Comprende sentencias Transact-SQL.
- Administra y accesa a muchas bases de datos.
- Optimiza la ejecución de consultas.
- Proporciona backup.
- Impone reglas de trabajo.

SYBASE SQL Server es un programa DBMS múltiusuario. La información es pasada del cliente al servidor a través de la red, usando una de varias APIs estándar.



Sybase proporciona los siguientes objetos de bases de datos para reforzar la integridad de los datos en el servidor:

1. **Rules** (reglas). Establece un dominio de valores permitidos para una columna dentro de una base de datos.
2. **Check** (Restricciones de revisión). También ofrece un dominio de valores (a partir de la versión 10). Es muy similar a las reglas, pero con dos diferencias significativas: la primera es que debe ser definida como parte de la sentencia create table o alter table y la segunda es que puede referirse a más de una columna en la tabla.
3. **Defaults** (por omisión). Provee un valor para una columna cuando no es provisto un valor durante un insert.
4. **Unique indexes** (Indices únicos). Requiere que una columna o conjunto de columnas dentro de una tabla sean únicos para esa tabla. Los índices son también usados para perfeccionar los query's.
5. **Primary Key** (Llave primaria). Permite identificar cual columna o columnas comprenden la llave primaria de una tabla. Estas restricciones son definidas como parte del create table o alter table, dichas columnas no podrán ser nulos y un índice único será creado automáticamente cuando el comando se ejecute.
6. ***Unique constraints*** *(Restricción única). Permite identificar cual columna* o combinación de columnas serán únicas para cada renglón en la tabla. Estas restricciones son definidas como parte del create table o alter table, dichas columnas no podrán ser nulos y un índice único será creado automáticamente cuando el comando se ejecute.
7. **Referential integrity** (Restricciones de Integridad referencial). Las restricciones refuerzan la integridad referencial entre las tablas relacionadas.

8. **Triggers** (Gatillos). Permite al administrador de la base de datos especificar un conjunto de instrucciones SQL para ser ejecutados después de cualquier modificación de una tabla: INSERT, UPDATE o DELETE. Los gatillos son usados para mantener la integridad referencial, para mantener la dependencia derivada de datos y para la ejecución de complejas validaciones de datos.
9. **Stored procedures** (Procedimientos almacenados) Habilita a usuarios autorizados definir rutinas SQL ejecutables para operar en el servidor, los procedimientos almacenados pueden ser especificados por nombre y aceptar parámetros, pueden mejorar considerablemente el desempeño del servidor, frecuentemente son usados para afianzar los estándares en las rutinas de modificación de tablas.

Con las herramientas anteriores se permite una integridad de los datos manejada dentro de las aplicaciones, resultando una mayor seguridad de los mismos.

## Tablas del sistema

La base de datos master contiene tablas del sistema que controlan la información sobre SQL Server de forma global. Además cada Base de Datos (incluida Master) contiene tablas de sistema que controlan información especifica.

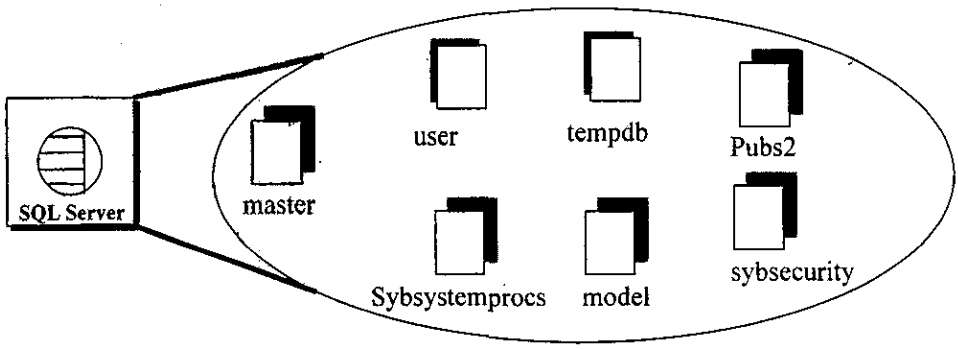
Cuando se instala SQL Server, se crea una Base de Datos Master. Las tablas de sistema de una Base de Datos de usuario se crean automáticamente cuando se ejecuta el comando *Create DataBase*.

**Advertencia sobre las tablas del sistema**: las tablas de sistema de SQL Server contienen información que es crucial para el funcionamiento de las bases de datos. Los datos de estas tablas se insertan, actualizan y eliminan mediante

comandos de Transact-SQL como Create y Drop o mediante procedimientos del sistema. En circunstancias normales no es necesario hacer modificaciones directas en los datos de las tablas del sistema.

Al instalar SQL Server se crean las siguientes Bases de Datos del sistema:

- Base de Datos Master.
- Base de Datos Model.
- Base de Datos de procedimientos del sistema, Sybsystemprocs.
- Base de Datos Temporal, tempdb.




**Base de Datos Master.** (5MB mínimo). Controla el funcionamiento de SQL Server en su conjunto y almacena información sobre todas las Bases de Datos de usuario y sus dispositivos asociados. Hace seguimiento de:

- Cuentas de usuarios (Syslogins).
- Cuentas de usuarios remotos (sysremotelogins).
- Servidores remotos con los que puede interactuar este servidor (syssevers).
- Procesos en curso (sysprocesses).
- Variables de entorno configurables (sysconfigures).
- Mensajes de error del sistema (sysmessages).

- Bases de Datos en SQL Server (sysdatabase).
- Cintas y discos montados en el sistema (sysdevices).
- Juego de caracteres (syscharsets).
- Usuarios que tienen roles en todo el servidor (Sysloginroles).

Dado que master guarda información sobre los dispositivos y bases de datos de usuario, es necesario estar en la base de datos master para ejecutar *create, alter, disk init, disk refit, disk reinit* * y los comandos de duplicación de disco.

**Base de Datos Model.** (2MB). Sirve de plantilla o prototipo de las nuevas bases de datos de usuarios. Cada vez que un usuario introduce el comando create database, SQL Server realiza una copia de la base de datos Model y extiende la copia al tamaño especificado por el comando create database.
**Nota**: Una Base de datos nueva no puede ser más pequeña que la Model.

**Base de datos tempdb.** (2 MB mínimo) SQL Server tiene una base de datos temporal, que proporciona un área de almacenamiento para tablas temporales y otras necesidades temporales (por ejemplo: resultados intermedios de group by y order by) el espacio de tempdb se comparte entre todos los usuarios de todas las bases de datos del servidor. El tamaño predeterminado de tempdb es de 2 MB. Se puede alterar su tamaño con el comando alter database, si es necesario.

**Base de datos sybsecurity.** Es una base de datos que contiene el sistema auditor de SQL Server (no se instala automáticamente, es necesario correr el programa de instalación), consta de la tabla sysaudits, que contiene la lista de auditoria y la tabla sysauditoptions que describe las opciones globales de auditoria.

* Lista de comandos de Transact-SQL en la página 137

**Base de datos pubs2.** (2 MB). La instalación es opcional, diseñada como utilidad de aprendizaje, pubs2 es la base en la mayoría de los ejemplos presentados en la documentación de SQL Server (Una base de datos de Prueba).

**Base de datos sybsystemprocs.** Contiene los procedimientos almacenados del sistema.

**Base de datos User.** Bases de datos definidas por el usuario

## COMANDOS PARA EL MANEJO DE SYBASE

La siguiente tabla describe algunos comandos de Transact-SQL

| Comandos | Descripción |
|---|---|
| **alter database** | Aumenta la cantidad de espacio asignado a una base de datos. |
| **alter table** | Añade nuevas columnas, cambia u omite restricciones en una tabla. |
| **create database** | Crea una base de datos. |
| **create index** | Crea un índice con una o más columnas de una tabla. |
| **create pocedure** | Crea un procedimiento almacenado que acepta uno o más parámetros suministrados por el usuario. |
| **craete rule** | Especifica el dominio de valores aceptables de una columna o de cualquier columna de un tipo de datos especifico. |
| **create table** | Crea nuevas tablas y restricciones de integridad opcionales. |
| **create trigger** | Crea un disparador. |
| **create view** | Crea una vista, la cual es una forma alternativa de ver los datos de una o más tablas. |
| **dbcc** | Es el verificador de consistencia lógica y física de bases de datos. |
| **delete** | Quita filas de una tabla. |
| **drop database** | Quita una o más bases de datos de un SQL Server. |
| **drop default** | Quita un valor predeterminado definido por el usuario. |

| | |
|---|---|
| **drop index** | Quita un índice de la tabla de la base de datos actual. |
| **drop procedure** | Quita procedimientos almacenados. |
| **drop rule** | Quita una regla definida por el usuario. |
| **drop table** | Quita la definición de una tabla y todos sus datos, índices, disparadores y especificaciones de permisos de la base de datos. |
| **drop trigger** | Quita un disparador. |
| **drop view** | Quita una vista de la base de datos. |
| **execute** | Ejecuta un procedimiento de sistema o uno almacenado definido por el usuario. |
| **grant** | Asigna permisos a los usuario |
| **insert** | Añade filas a una tabla o vista. |
| **load database** | Carga una copia de seguridad de una base de datos de usuario incluido su diario de transacciones. |
| **load transaction** | Carga una copia de seguridad del diario de transacciones. |
| **revoke** | Elimina permisos de los usuario. |
| **rollback** | Revierte una transacción definida por el usuario al último punto de resguardo dentro de la transacción. |
| **rollback trigger** | Revierte el trabajo realizado por un disparador, incluyendo la actualización que provoco la activación del disparador y ejecuta una instrucción opcional raiserror. |
| **select** | Recupera filas de los objetos de la base de datos. |
| **shutdown** | Cierra SQL Server o un Backup Server. Sólo el administrador puede ejecutar este comando. |
| **truncate table** | Quita todas las filas de una tabla. |
| **update** | Cambia datos de la filas. |
| **Use** | Especifica la base de datos con la que se desea trabajar. |

**Nota:** Casi todas las tareas de administración del sistema requieren que se conecten a SQL Server con la utilidad isql.[2]

[2] **isql** (Interactive SQL) Es la forma interactiva genérica de SQL para SQL Server, frecuentemente es referenciado como un trabajo con líneas de comando, isql es usado para ejecutar actividades tales como: crear objetos, probar, insertar datos, y seleccionar información. Isql usualmente corre como una colección de comandos en línea.

## TIPOS DE DATOS

Los tipos de datos especifican el tipo de información, tamaño y formato de almacenamiento de las columnas, parámetro de procedimientos almacenados y variables locales.

| Tipo de Dato | Bytes de Almacenamiento | Rango | Ejemplo |
|---|---|---|---|
| **Numéricos** | | | |
| Tinyint | 1 | 0 a 255 | 200 |
| Smallint | 2 | $2^{15}$ -1 (32767) a -$2^{15}$(-32768) | 23447 |
| Int | 4 | $2^{31}$(2147483647) a - | 476887 |
| Numeric (p,s) | 2 a 17 | $2^{31}$(2147483648) | 55.903 |
| decimal(p,s) | 2 a 17 | $10^{38}$ -1 a $-10^{38}$ | -992.34 |
| | | $10^{38}$ -1 a -$10^{38}$ | |
| **Aproximaciones** | | | |
| Float(precisión) | 4 a 8 | Depende de la maquina | 3.243 |
| double | 8 | Depende de la maquina | |
| real | 4 | Depende de la maquina | |
| **Dinero** | | | |
| smallmoney | 4 | 214748.3647 a -214748.3648 | $ 48.95 |
| *money* | 8 | *922337203685477.5807 a* | |
| | | -922337203685477.5808 | |
| **Fecha/Hora** | | | |
| smalldatetime | 4 | Enero/1/1900 a Junio/6/2079 | "3-19-93" |
| datetime | 8 | Enero/1/1753 a Diciembre/31/9999 | |
| **Caracter** | | | |
| char(n) | | | |
| varchar(n) | n | 255 caracteres o menos | "RB CR" |
| nchar(n) | Longitud de la cadena | 255 caracteres o menos | |
| nvarchar(n) | n* @ @charsize | 255 bytes o menos | |
| | @ @ncharsize * | 255 bytes o menos | |
| text | numero de caracteres | | |
| | 0 o múltiplos de 2 K | $2^{31}$-1(2147483647) bytes o menos | |
| **Binario** | | | |
| binary(n) | n | 255 bytes o menos | =xf3cf |
| varbinary | Longitud de entrada | 255 bytes o menos | |
| image | 0 o múltiplos de 2K | $2^{31}$-1(2147483647) bytes o menos | |
| **Bit** | | | |
| | 1 byte almacena hasta 8 | 0 o 1 | 1 |
| bit | bits | | |

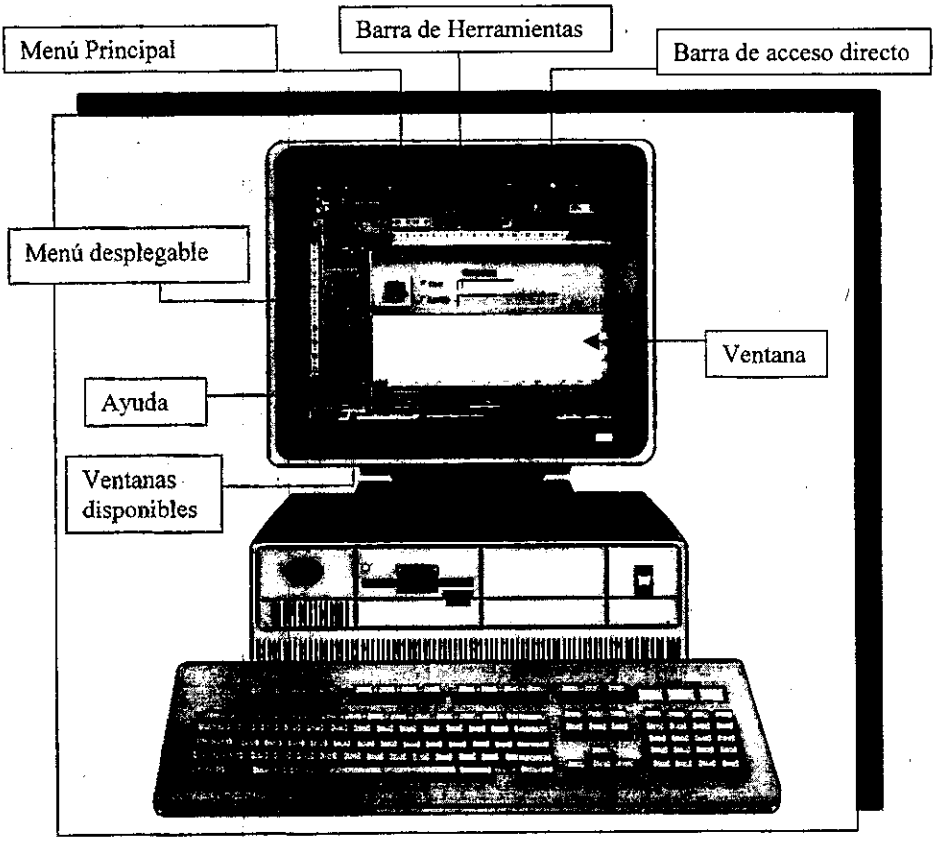
**Nota:** SQL Server proporciona tipos de datos de sistema y tipos de datos definidos por el usuario (timestamp y sysname).

# IV.3 INTERFACE GRAFICA DE USUARIO (GUI)

La interfaz de usuario es la puerta hacia aplicaciones de software interactivas. A medida que el hardware se ha hecho más eficiente y se ha aprendido más acerca de los factores humanos y su impacto en el diseño de la interfaz, han ido apareciendo modernas interfaces orientadas a ventanas con opción de señalar y elegir, las cuales ofrecen grandes ventajas como:

- Se pueden visualizar diferentes tipos de información simultáneamente, permitiendo al usuario cambiar de contexto (escribir en una ventana, examinar resultados en otra) sin perder conexión visual con otros trabajos.
- El esquema de menús desplegables permite realizar muchas tareas interactivas diferentes, ya que permiten al usuario llevar a cabo tareas de control y diálogo de forma sencilla.
- La utilización de iconos gráficos, menús desplegables, botones y técnicas de presentación continua reducen el número de pulsaciones en el teclado, y por consiguiente reduce los errores en las entradas, minimizando el número de validaciones de datos provocando un aumento en la eficiencia del sistema.

Se ha comprobado que la comunicación visual tiene la cualidad del *"paralelismo"*, Ya que presenta simultáneamente muchos elementos de información distintos por asimilar, pero sencillos de comprender con la ayuda de gráficas relacionadas al proceso que realiza.




No hay duda que por ello, la tendencia actual son las interfaces con multitarea, ventanas y dispositivos de selección, sin embargo sólo funcionará correctamente si se lleva a cabo un diseño cuidadoso de esta, dado que la especificación apropiada de la comunicación visual es el elemento clave de una interfaz **"amigable".**

Finalmente, la información, según se extrae de la interfaz, debe ser almacenada para ser usada posteriormente.

# POWERBUILDER (PB) v. 5

Una interface gráfica de usuario es un entorno orientado a objetos, dirigido a eventos. Las ventanas, así como los controles y otros objetos que aparecen en ellas son objetos, y los usuarios pueden provocar varios eventos (hacer clic en el ratón sobre la ventana) y los programas realizados con PB responderán a estos eventos, el comienzo de la respuesta al evento controla la operatibilidad de la aplicación. PB trabaja sobre Windows 3.11, Windows 95, Windows NT, Macintosh y UNIX.

## Herramientas

- Cuenta con un conjunto de herramientas conocidas como ***painters,*** donde cada painter soporta una funcionalidad distinta, con un propósito particular.

- Cuenta con el lenguaje **PowerScipt** para la creación de las aplicaciones.

- Un objeto propio llamado **Objeto Data Windows**, que es usado para la manipulación de datos y reportes.

- Soporta todos los VBX 1.0, OCX y los controles de Windows DDE (Dinamic data Enchange), DLL (Dinamic Link Librarie), OLE 2.0 (Objet Link and Enbedding)

**Métodos.** Son los que controlan a los objetos, existiendo dos tipos de métodos:

**EVENTOS:**

- Cada objeto cuenta con un conjunto de eventos predefinidos, por ejemplo: Clicked, Constructor, Open, etc.
- Se le proporciona código script al evento para que efectúe el proceso indicado.

**FUNCIONES:**

- PB incluye un conjunto de funciones de sistema tales como Open( ) y close( ), que están disponibles para todos los objetos.
- Cada objeto llega a tener también su propio conjunto predefinido de funciones. Por ejemplo: el objeto Data Window tiene la función Retrieve( ) y Update( ).

PB crea y modifica objetos usando herramientas gráficas llamadas painters, el acceso a los painters es a través de la tool bar (barra de herramientas) principal, conocida como Power Bar.

Algunos painter se presentan a continuación.

| Icono del Painter | Nombre del Painter | Descripción |
|---|---|---|
|  | Aplication | Es un elemento no visual pero es el punto de entrada dentro de una colección organizada de ventanas y otros objetos que definen la aplicación. |
|  | Window | Son objetos visuales que proporcionan la interface principal entre el usuario y la aplicación. |
|  | Menú | Objeto visual que proporciona una lista de comandos, opciones o alternativas para ejecutar una tarea. |

| | | |
|---|---|---|
|  | Data Window | Combina el acceso a los datos con la interface gráfica de usuario Nota: El Data Window painter genera un objeto Data Window, el Window painter crea un control Data Window. Este control puede ser asociado a un objeto Data Window especifico. |
|  | Structure | Es una colección de uno o más grupos de variables relacionadas bajo un mismo nombre. Permite generar nuevas estructuras o modificar una ya existente(añadir elementos o cambiarlos). |
|  | Help | Abre la ayuda en línea de PB. |
|  | Table | Permite crear, modificar y eliminar tablas, llaves primarias y llaves foráneas. |
|  | Database | Suministra facilidades para el mantenimiento de la base de datos tales como la definición y el acceso de datos, añadir o borrar tablas de una base de datos o cambiar su formato, modificar datos. El painter Base de Datos es donde se ejecuta el trabajo con la base de datos en PB. También es el lugar donde los otros painters relacionan diferentes bases de datos. |
|  | Pipeline | Copia las tablas de datos de una base de datos a otra. |
|  | Query | Se definen sentencias SQL que pueden ser copiadas en un objeto Data Window para el código fuente. |
|  | Funtion | Consiste en un conjunto de sentencias de PowerScipt que ejecutan un tipo de proceso y regresa un valor. |
|  | Project | El painter proyect permite crear una definición de proyecto |

| | | |
|---|---|---|
|  | Library | Una librería incluye objetos o controles de PB donde una aplicación esta formada por una o varias librerías (archivos con extensión .pbl). El painter de librerías suministra todas las facilidades necesarias para la gestión de las librerías y de los objetos que hay en ellas. |
|  | User Objects | Son objetos personalizados que crea el usuario, generalmente consisten en uno o más objetos estándares de PB con script para eventos particulares. |
|  | Run | Permite correr la aplicación. |
|  | Debug | Permite colocar puntos de ruptura, el correr la aplicación una sentencia a la vez (paso a paso) y examinar variables durante la ejecución. |

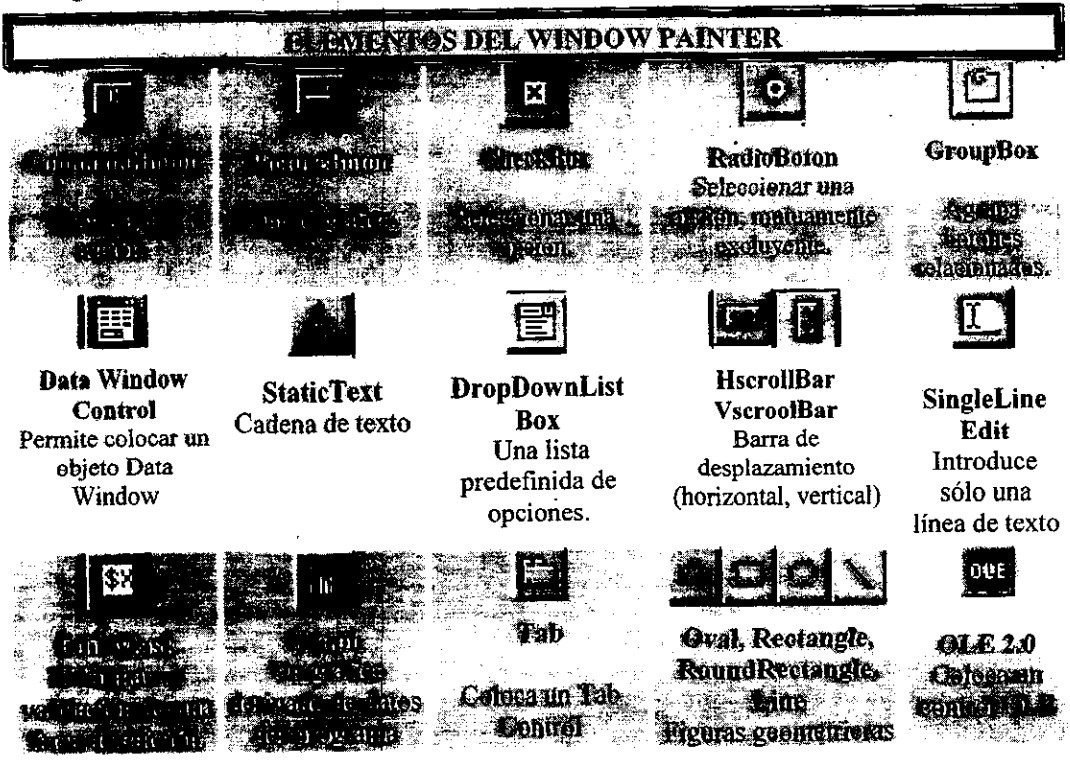
## Creando una ventana.

Una ventana es un objeto visual que proporciona la interface principal entre el usuario y la aplicación PB. PB maneja cuatro tipos de ventana cada una con sus características (propiedades) y conducta (métodos) apropiada para situaciones concretas:

- **Main** (Principal): Primera ventana que aparece en una aplicación y opera independientemente.
- **Child** (Hija): Ventana secundaria respecto a una ventana padre y solo existe dentro del marco de su ventana padre.
- **Popup** (Emergente): Normalmente muestra información, como por ejemplo ayuda en línea.
- **Response** (Respuesta): Muestra un mensaje de error o aviso y requiere de una respuesta del usuario.

- **MDI[3] frame**: Es una ventana que permite habilitar varias ventanas dentro de su frame llamadas hojas y que puede contener múltiples instancias iguales o diferentes.
- **MDI frame with microhelp**: MDI frame y MDI frame with microhelp son básicamente iguales, la diferencia radica en que la segunda tiene la característica de soportar un número limitado de palabras que pueden aparecer dentro de la barra de status.

A una ventana se le pueden añadir diferentes objetos estándares de Windows, controles PB especializados o definidos por el usuario. A continuación se presentan algunos de estos objetos.




[3] **MDI (Multiple document interface).** Es un tipo de aplicación para manejar múltiples ventanas iguales o de diferente tipo. Las aplicaciones MDI son disponibles en Windows y UNIX.

## PowerScript

PowerScript es el lenguaje de programación propio de PB, es utilizado para escribir los procedimientos (script). Un script es la respuesta codificada a un evento por lo que cuando un evento ocurre PB ejecuta el script de este evento. Los procedimientos pueden incluir sentencias ordenes y funciones.

| Tipos de Sentencia | Descripción |
|---|---|
| Declaración de variables. | Es la asignación de un identificador con respecto a un tipo de dato establecido por PB. |
| Llamadas a función. | Es la llamada a un grupo predefinido de sentencias, identificadas por un único nombre que realiza una tarea especifica. La función puede ser llamada por varios procedimientos. (PB proporciona una variedad de funciones). |
| Sentencias de asignación. | Es la acción de almacenar un determinado valor a alguna variable, objeto u propiedad de objeto. |
| Sentencia de control de flujo. | Permiten controlar el orden de ejecución de las sentencias de un procedimiento:<br>• If Then Else<br>• Choose Case<br>• Do Loop<br>• For Next<br>• Goto<br>• Halt<br>• Return |
| Sentencias embebidas SQL. | Se incluyen sentencias SQL que el procedimiento utiliza para pedir un servicio a la Base de Datos. |

Sin embargo, para poder hacer uso de powerscript, es importante tener en mente las palabras reservadas, identificadores y tipos de datos que maneja.

IDENTIFICADORES Y TIPOS DE DATOS:

Los identificadores permiten reconocer los objetos mediante la asignación de nombres que deben empezar con una letra y utilizar como máximo 40 caracteres. Sin embargo existen ciertas palabras que no pueden ser utilizadas como identificadores ya que PB las utiliza con una determinada finalidad. A continuación se listan algunas:

| | | | |
|---|---|---|---|
| ***and*** | ***disconnect*** | ***if*** | ***return*** |
| ***call*** | ***to*** | ***insert*** | ***rollback*** |
| ***case*** | ***else*** | ***into*** | ***select*** |
| ***choose*** | ***elseif*** | ***is*** | ***system*** |
| ***close*** | ***end*** | ***loop*** | ***this*** |
| ***commit*** | ***execute*** | ***next*** | ***true*** |
| ***connect*** | ***false*** | ***open*** | ***update*** |
| ***create*** | ***for*** | ***private*** | ***using*** |
| ***delete*** | ***goto*** | ***protected*** | ***while*** |
| ***destroy*** | ***halt*** | ***public*** | |

## TIPOS DE DATOS (Powerscript)

| Tipo | Descripción |
|---|---|
| Any | Asume el tipo de dato del valor asignado (valores simples, estructuras, objetos, arrays, etc.). |
| Blob | Dato sin formato. |
| Boolean | Verdadero o Falso. |
| Char | Un solo carácter. |
| Date | Fecha con el formato yyyy-mm-dd. |
| Datetime | Combina la fecha y la hora en una sola variable. |
| Double | Números con punto flotante, 15 dígitos de precisión (su rango es de 2.2 E –308 a 1.7 E +308). |

| | |
|---|---|
| Decimal | Número decimal con 18 dígitos de precisión. |
| Entero | Enteros con signo de 16 bits (rango –32768 a 32767). |
| Long | Enteros con signo de 32 bits (rango –2,147,483,648 a 2,147,483,647). |
| Real | Numero con punto flotante con precisión de 6 dígitos (rango 1.17 E-38 a 3.4 E+38) |
| String | Puede contener desde 0 hasta 60,000 caracteres ANCII. |
| Time | Tiempo en el formato HH:MM:SS. |
| Unsignedinteger | Entero sin signo de 16 bits de 0 a 55535. |
| Unsignedlong | Entero largo sin signo de 32 bits de 0 a 4,294,967,295. |

El alcance de las variables determina desde que parte de una aplicación pueden ser utilizadas, pudiendo ser:

**Global**.- Accesible desde cualquier parte de una aplicación.

**Instancia**.- Corresponden a una instancia de un objeto.

**Locales**.- Solo en el procedimiento se pueden referenciar a las variable locales que se definan.

**Shared**.- Una clase de variable accesible para todos los script por todas las instancias de las clases de objeto para la cual la variable es declarada y para cualquier objeto descendente.

Algunos operadores permiten realizar operaciones aritméticas (suma, resta, exponenciación, etc.), los operadores lógicos permiten hacer operaciones sobre valores buleanos mientras que otros operadores permiten la manipulación de cadenas.

| | |
|---|---|
| **Aritméticos:** | + - * / ^ ++ += -- -= *= /= ^= |
| **Relacionales:** | = < > <= >= <> |
| **Lógicos:** | And Or Not |
| **Concatenación de cadenas:** | + |

## Objeto Transacción

Un DBMS es responsable de la integridad de los datos. Las transacciones son soportadas en el lenguaje Power Script, donde una aplicación necesita un objeto diferente para cada conexión de Base de Datos. El objeto transacción por defecto en PB es SQLCA. Para utilizar múltiples conexiones a Bases de Datos en una aplicación se siguen los siguientes pasos:

1. Definir el objeto transacción: *Transaction Conefesz.*

2. Instanciar el objeto transacción: *Conefesz = Create Transaction.*

3. Inicializar el objeto transacción:

   *Conefesz.DBMS = "Sybase"*
   *Conefesz.DataBase = "Conefesz"*
   *Conefesz.Logid = "RB"*
   *Conefesz.Logpass = "dM23"*
   *Conefesz.Servername = "Zaragoza"*
   *Conefesz.Autocommit = false*

Se debe destruir el objeto transacción cuando ya no sea necesario para liberar recursos: *Destroy Conefesz*

4. Conectar a una Base de Datos: *Connect Using Conefesz*[4];

Para determinar si la conexión tubo éxito o no, se debe incluir lo siguiente:

```
If conefesz.SQLcode < 0 then      //no se conecto
    ...bloque de sentencias
    halt close                    //cierra la aplicación
End if
```

5. Deshaciendo la conexión: *Disconnect Using Conefesz*[4];

## Ejemplos de codificación de procesos de SABZ con Power Script.

### CONEXIÓN A LA BASE DE DATOS

**Evento: clicked Objeto Cb_aceptar ventana: w_password**

```
string ls_cveusuario
ls_cveusuario = sle_password.text /* Se toma el dato del objeto SingleLineEdit
SELECT "usuarios"."cveusu"  /* Selecciona el Usuario de la Tabla de Usuario */
 INTO :gs_usuario /* variable global */
 FROM "usuarios"
 WHERE "usuarios"."cveusu" = :ls_cveusuario
 USING sqlca;
IF len(trim(gs_usuario)) > 0 Then     // solo si existe clave
  CHOOSE CASE Upper (gs_usuario)  // solo para los que tenga acceso a Conefesz
      CASE "SYBASE"    // Transaction Object a la Base de Datos dbconefez
            dbconefesz = CREATE transaction
            dbconefesz.dbms = "ODBC"
            dbconefesz.dbparm = Connectstring='DSN=Conefesz;uid=DBA;pwd=sql'"
            CONNECT USING dbconefesz;
              IF dbconefesz.SQLcode <> 0 THEN
                 MessageBox("Mala Coneccion", "Imposible la coneccion a &
                 Conefesz " + dbconefesz.SQLerrText)
                Return
              END IF
  END CHOOSE
```

[4] Las instrucciones Connect y disconnect son sentencias SQL embebidas, y deben finalizar con punto y coma.

```
// Estas sentencias las realiza con o sin coneccion a conefesz
 open (w_mdiprincipal)
 close (parent)
Else
  Messagebox("Usuario", "Password Erroneo")
     // proceso de error en Password, manda sentellar al grafico
     beep(3)
     timer(0.5)
     sle_password.setFocus()
End IF
```

## ALTAS DE LECTORES

**Evento: clicked  Objeto: Cb_salvar Ventana: w_lectores**

```
IF dw_consulta.update() = 1 THEN              //actualiza los cambios en la BD
  COMMIT using SQLCA;
   IF m_lectores.m_movlectores.m_lec_altas.checked THEN
      inserta_nuevo_renglon ( ) //llamada a función
   END IF
ELSEIF lb_consulta then
  ROLLBACK using SQLCA;
  MessageBox ("FALLA", "No se ha podido guardar ningun dato,consulte"&
            + " con su administrador de red", Information!, OK!)
 Return
END IF
ID  dw_lec_plus.update() = 1 THEN
  COMMIT using SQLCA;
ELSEIF lb_plus then
  ROLLBACK using SQLCA;
  MessageBox("FALLA", "No se ha podido guardar ningun dato,consulte"&
            + " con su administrador de red", Information!, OK!); Return
END IF
```

## BAJAS DE LECTORES

**Evento: clicked Objeto: Cb_bajas Ventana: w_lectores**

```
Int li_respuesta
li_respuesta = MessageBox("DAR DE BAJA AL LECTOR", &
"¿ESTA SEGURO DE QUERER BORRA AL LECTOR?",Question!,YesNo!,2)
IF li_respuesta = 1 THEN
   dw_consulta.DeleteRow(0)
   IF dw_consulta.update() = 1 THEN
      COMMIT using SQLCA;
  ELSE
      ROLLBACK using SQLCA;
      MessageBox("FALLA", "Error en la base de datos;No se ha eliminado" &
        + " al lector, consultar con su administrador de red", Information!, OK!)
      Return
  END IF
       dw_lec_plus.DeleteRow(0)
ELSE // cancela operación de borrado
       m_lectores.m_movlectores.m_lec_bajas.triggerevent ( clicked! )
END IF
```

## DESCONEXIÓN DE LA BASE DE DATOS

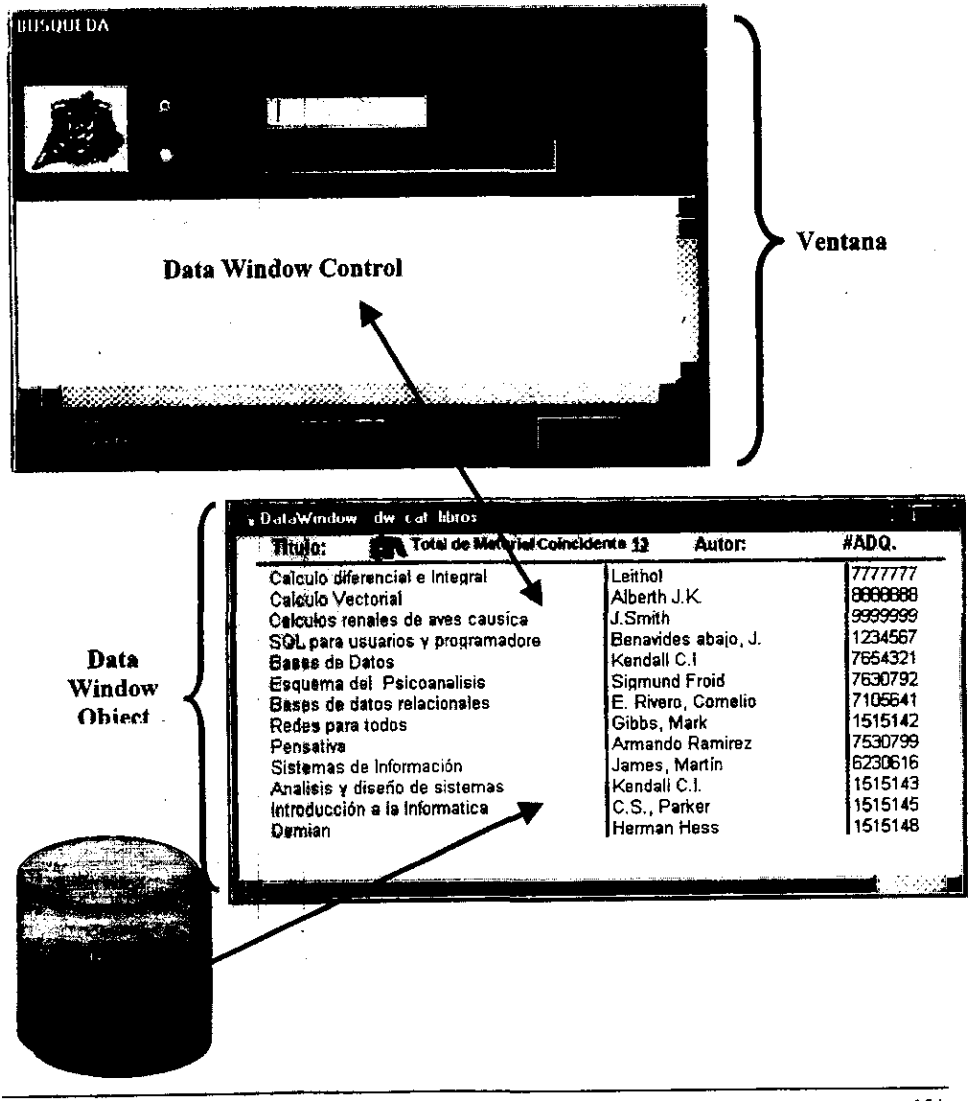
**Evento: close Objeto: aplicación sabz**

```
SetPointer(HourGlass!)     //Desconeccion a las bases de datos

DISSCONECT using SQLCA;
IF sqlca.sqlcode < 0 THEN
    MessageBox("Error",  "No es posible dar de baja la base de
    datos",Information!)
ELSE
    MessageBox("Desconexión",  "Ya no existe conexión con la base de datos.)
END IF
CHOOSE CASE gs_usuario
       CASE "sybase"
       DISCONNECT using DBCONEFESZ;
END CHOOSE
```

## Objeto DataWindow

Un objeto DataWindow (ventana de datos) presenta, manipula, actualiza e imprime informes de datos. Este objeto único de PB encapsula información acerca de una fuente de datos, incluye el formato de la información y describe como los datos serán desplegados en un control DataWindow (colocado en alguna ventana).



La ejecución de una aplicación comienza con la rutina de carga y arranque inicial contenida en un archivo.exe. Esta rutina hace uso del "Powerbuilder DataBase Development and Deployment Kit" (DDDK), Kit de distribución y desarrollo de Powerbuilder. El DDDK contiene una serie de programas que permiten a Windows acceder a las versiones compiladas de los objetos contenidos en el archivo ejecutable. Por lo que, al distribuir una aplicación PB, además del archivo ejecutable el usuario requiere del DDDK y también en caso de existir, todas las librerías que contienen la aplicación.

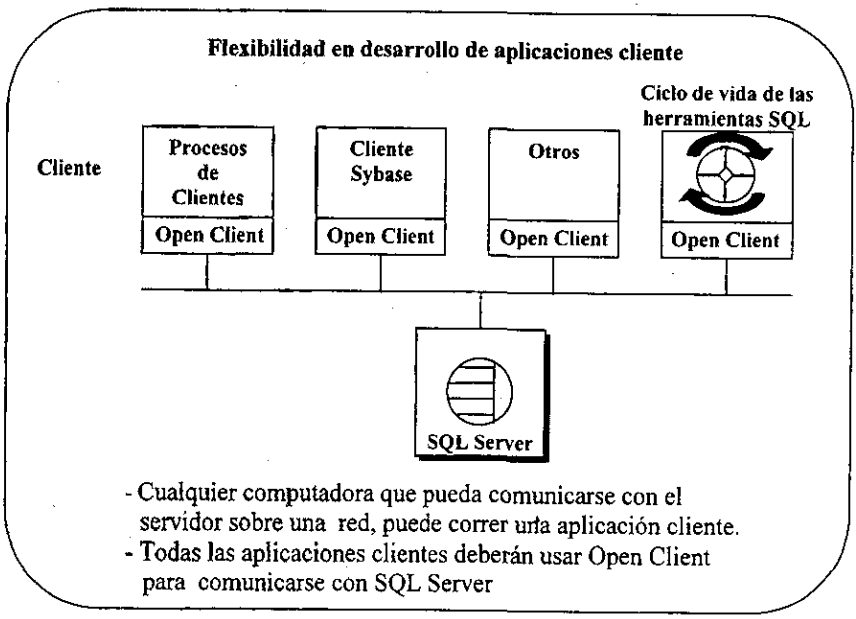
PB tiene todas las herramientas necesarias para construir aplicaciones Cliente/Servidor. Permite realizar aplicaciones que se ejecutan en la parte del cliente, comunicándose entre sí mediante protocolos de red estándar. Las aplicaciones pueden acceder a datos locales o remotos provenientes de una serie de fuentes. En el servidor se pueden ejecutar la mayor parte de los sistemas de bases de datos, pudiendo ser computadoras personales, estaciones de trabajo o minicomputadoras.

## IV.4 CONECTIVIDAD.

### OPEN CLIENT

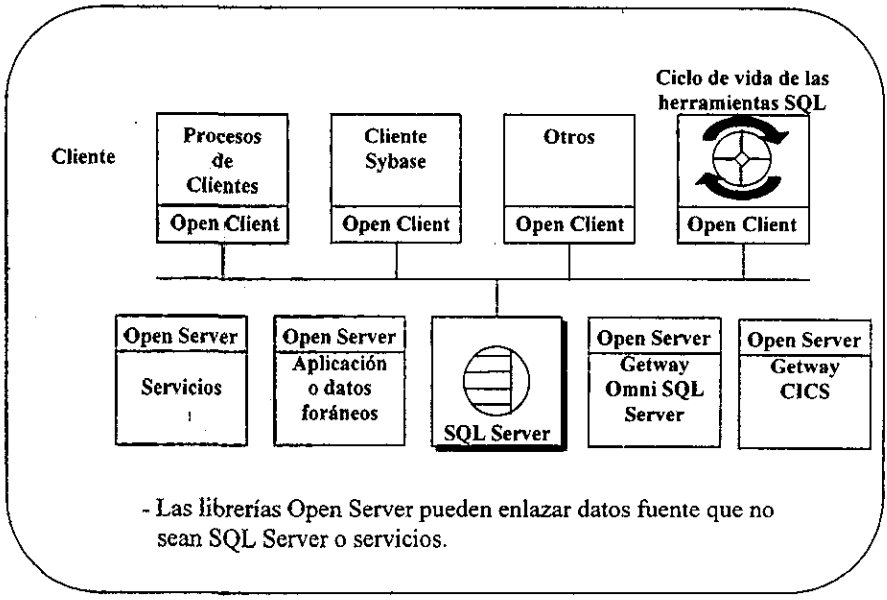
Open Client es un producto que permite la conectividad con Sybase. Es una colección de librerías. Una librería es una colección de rutinas estándar o llamadas, que son adicionadas a un programa de aplicación. Sybase Open Client es requerido en las aplicaciones cliente para enviar peticiones a SQL Server. Con Open Client una API (Aplication Programing Interface) se hace disponible para la aplicación cliente permitiendo la comunicación con SQL Server.

Todos los clientes requieren Open Client para comunicarse con un servidor SQL. Para varias plataformas los manejadores de red de los protocolos de red estándar están incluidos en Open Client. La plataforma determina los componentes de Open Client que son requeridos para las necesidades computacionales Cliente/Servidor. El servidor también contiene manejadores de red similares para establecer la comunicación con el cliente.




## OPEN SERVER

Open Server suministra una interface de programación flexible, que permite cualquier fuente de datos o servicios de sistema o aplicaciones de información para responder a las peticiones de usuario como si residieran en un SQL Server,




Ambos, Cliente y Servidor usando estas librerías pueden comunicarse en un ambiente computacional heterogéneo, esto es, un cliente vía Open Server puede comunicarse con otro DBMS u otra fuente de datos. Un sistema creado con Open Server permite a operaciones basadas con Open Client accesar a un amplio rango de datos y servicios distribuidos a través de diferentes plataformas.

# IV.5 EVALUACIÓN, CAPACITACIÓN, MANTENIMIENTO Y SEGURIDAD.

## Evaluación.

A todo lo largo del ciclo de vida del sistema, todos los involucrados en el desarrollo del sistema (desarrolladores, analistas, directivos, usuarios finales, etc.) deberán evaluar la evolución del sistema con el fin de generar una retroalimentación que permita su perfeccionamiento.

Se disponen de muchas técnicas diferentes de evaluación, una manera directa para evaluar un nuevo sistema son las utilerias de información que corresponden y contestan las preguntas de Quién (Posesión), Qué (Forma), Dónde (Lugar), Cuándo (Tiempo), Cómo (Actualización) y Porqué (Objetivo).

- **Posesión.** ¿Quién debe recibir la salida? ó ¿Quién toma las decisiones?. La información carece de valor en las manos de alguien que carece de la habilidad para utilizarla.
- **Forma.** ¿Qué tipo de salida se distribuye?. El documento debe ser útil, respetando el formato y lenguaje que se haya definido.
- **Lugar.** ¿Dónde debe distribuirse la información?. La información debe llevarse al sitio que la demande.
- **Tiempo.** ¿Cuándo debe proporcionarse la información?. La información debe llegar anticipadamente al momento de la decisión. La información tardía no tiene valor, los reportes suelen ser poco precisos u olvidarse si se entregan prematuramente. La información, por tanto debe llegar en el momento justo.

- **Actualización.** ¿Cómo se introduce la información y Cómo se utiliza?. Que la información sea valida, real y actual.
- **Objetivo.** ¿Porqué el sistema?. Evaluar si las salidas tienen valor para auxiliar a la organización a alcanzar sus objetivos.

Un sistema de información puede clasificarse como exitoso si posee las 6 utilerias anteriores.

## Capacitación:

Incluso los sistemas bien diseñados y técnicamente eficientes pueden tener éxito o fallar debido a la forma en que se operan y se utilizan, por lo tanto, la calidad de la capacitación del personal involucrado con el sistema en varias de sus facetas, ayuda o dificulta y puede incluso obstaculizar por entero la puesta en marcha de un sistema de información. Las personas que trabajarán con el sistemas o que se verán involucradas por éste (ya sea para proporcionar información, recibir información o de hecho para operar el equipo) deben conocer a detalle las funciones que desempeñará, lo que no hará y sobre todo el cómo utilizarlo. Tanto los operadores como los usuarios finales necesitan capacitación.

*La capacitación para los operadores de los sistemas incluye no sólo el cómo* utilizar el equipo, sino también cómo diagnosticar las fallas de funcionamiento y que pasos son necesarios realizar cuando estas ocurran.

La capacitación incluye también la instrucción en los procedimientos de corrida del sistema y las actividades normales de operación; por ejemplo la carga de archivos, el cambio de formas de la impresora, el inicio y termino de la comunicación de datos.

La mayor parte de la capacitación de los usuarios se enfoca a la operación del sistema mismo, concentrando la atención en los procedimientos de manejo de datos. Es importante que los usuarios reciban la capacitación adecuada para los métodos de añadir transacciones, editar datos, formular consultas y borrar registros. Ninguna capacitación esta compuesta sin la familiarización con las actividades sencillas de mantenimiento del sistema. Las deficiencias de cualquier aspecto de la capacitación posiblemente llevarían a situaciones difíciles que producirán frustraciones y errores de usuarios.

Durante las sesiones de enseñanza se describe el nuevo sistema, se introduce la nueva tecnología incorporada a el y se analizan las ventajas esperadas sobre el sistema antiguo.

## Mantenimiento.

Después de la aceptación del nuevo sistema, comienza la parte del mantenimiento del ciclo de vida, el sistema se mantiene mediante cambios menores o mayores según sea necesario.

La fase de mantenimiento se centra en el ***cambio*** que va asociado a la corrección de errores, a las adaptaciones requeridas por la evolución del entorno del software y a las modificaciones debidas a los cambios de los requisitos del cliente dirigidos para reforzar o ampliar el sistema. Durante la fase de mantenimiento se encuentran tres tipos de cambios:

1. Corrección. Cambios en el software para corregir los defectos.
2. Adaptación. Con el paso del tiempo es posible que cambie el entorno original para el que se desarrollo el sistema, por lo que el software requerirá cambios para adaptarse a su entorno externo.

3. Mejora. Conforme se utilice el software, el cliente/usuario puede descubrir funciones adicionales que podrían ser de utilidad si se incorporan al sistema.

Se puede apreciar, que el mantenimiento del sistema es, mucho más que una "corrección de errores", ya que abarca desde los errores latentes inherentes a un gran sistema (dado, que no es razonable asumir que las pruebas del sistema hayan descubierto todos los errores), el rápido cambio que conlleva cualquier aspecto de la informática (Hardware y Software), hasta el proporcionar una mejor base para futuras modificaciones.

## Seguridad.

La seguridad de los servicios del sistema, así como la seguridad de los datos almacenados y de la información generada forman parte de una implantación eficiente.

Se deben considerar dos aspectos básicos con respecto a la seguridad:

**La seguridad del software** es una actividad que se centra en la identificación y evaluación de los riesgos potenciales que pueden producir un impacto negativo en el sistema y hacer que falle parcial o totalmente, ya sea por fallas de software o de Hardware. La seguridad del sistema examina los modos según los cuales los fallos producen condiciones que pueden generar accidentes.

**La seguridad en cuanto a Bases de datos**, significa permitir sólo a personal autorizados llevar a cabo acciones autorizadas sobre objetos especificados, sujetos a cualquier limitante administrativa. Una vez establecidos los permisos, muchos derechos de acceso y de procesos serán impuestos por el DBMS, esto es, deben

existir los mecanismos apropiados para asignar, controlar y revocar los derechos de acceso (leer, insertar, borrar o modificar) de cualquier usuario a cualquier dato o subconjunto de datos definidos en la base de datos. Al aumentar la cantidad de datos compartidos y el número de usuarios, aumenta la tarea del DBMS para garantizar tal seguridad. Una pieza de información o dato elemental debe protegerse completamente de intromisión no autorizada, ya sea accidental o intencionada.

# IV.6 CONCLUSIONES

Todo sistema de información transita por las etapas del ciclo de vida de un sistema. Aunque no sea tan evidente el paso de una etapa a otra, aquel sistema digno de ser considerado como tal, debe tener detrás de sí todo un análisis, diseño y desarrollo a detalle de las partes que lo conforman, para su perfecto funcionamiento.

Un sistema a la medida para la facultad es por mucho, mejor que un sistema estándar (SABZ Sistema automatizado de Biblioteca Zaragoza), debido a que al pretender abarcar y funcionar bajo todo tipo de situación, no cubren algunas particularidades propias de cada institución. El sistema a la medida, nace a partir de una problemática, para cubrir ciertas necesidades, crece y se desarrolla bajo estas cláusulas y da como resultado un sistema tal, que cubre los objetivos, requerimientos y perspectivas.

Partiendo de esta base, se impulsa la implantación de SABZ, con el objetivo de automatizar todos los procesos en que se involucre información que se genera la biblioteca.

Estamos convencidos, que el funcionamiento de SABZ, simplificará los tramites y procesos a realizar, tanto a los lectores como a todo el personal de la biblioteca. Seguros que la redundancia de información, falta de la misma, atrasos, etc., serán eliminados y que por otra parte, los tiempos de respuesta, así como la toma de decisiones serán de manera eficiente.

Se planea, que con la conformación del sistema Integral de Información de la Facultad de Estudios Superiores Zaragoza (CONEFESZ -Control Escolar de La

FES Zaragoza - y de SABZ), se logre que la información generada en los diferentes puntos neurales de la institución se comparta de manera eficiente y apoye al proceso de excelencia en los servicios.

# APENDICE A

# SQL

Este apéndice expone, definiciones básicas y describe el contexto en el que se usa el lenguaje SQL; proporcionando una perspectiva de lo que pretende y para que se utiliza.

El SQL es un lenguaje que permite expresar operaciones diversas, por ejemplo aritméticas, combinatoria y lógicas, con datos almacenados en Bases de Datos relacionales, que son aquéllas que se caracterizan porque la información está contenida en estructuras, llamadas tablas, donde los datos están dispuestos en filas y columnas.

La palabra SQL está formada con las iniciales de las palabras inglesas Structured Query Language, es decir, Lenguaje Estructurado de Consultas. El desarrollo de SQL empezó en los laboratorios IBM a mediados de 1970, dando

lugar a un lenguaje llamado SEQUEL. Se emitieron posteriormente varias versiones de SEQUEL, y para 1980 el producto se rebautizó como SQL. En los últimos años han ido apareciendo en el mercado múltiples productos de bases de datos basados en SQL, tanto para micro y miniordenadores como para grandes sistemas, haciendo que sea un lenguaje ampliamente utilizado en diversas máquinas y sistemas operativos. El Instituto Americano de Normas (American National Standards Institute ANSI) se ha asignado el papel de mantener a SQL, publicando de manera periódica versiones actualizadas del estándar.

Unas de las características más importantes del SQL frente a los lenguajes tradicionales no relacionales de bases de datos es que:

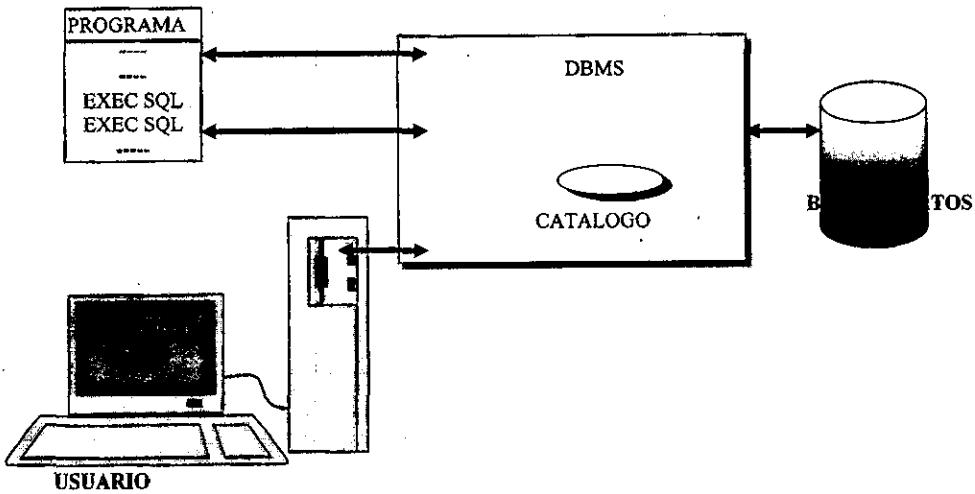
- Las sentencias permiten manejar conjuntos de registros en lugar de uno.
- Pose gran capacidad expresiva, aunque su estructura es simple, lo que permite expresar en una sola sentencia consultas complejas.
- Alta productividad en la codificación de programas.
- SQL permite definir y destruir objetos de la Base de datos.
- Conceder y denegar autoridad para usar estos objetos.
- Consultar y actualizar datos.

Los comandos SQL pueden ser utilizados en forma interactiva, como lenguaje de consulta o pueden insertarse en programas de aplicación, en este caso son procesados por un precompilador, por lo que SQL no es lenguaje de programación; más bien se trata de un ***sublenguaje de datos o un lenguaje de acceso a datos***, insertado en otros lenguajes.

Existen dos técnicas para utilizar el SQL embebido en programas, una de ellas llamada SQL estático es donde las sentencias están incluidas en el programa y no

pueden cambiar durante su ejecución, por lo que es mucho más sencillo y eficiente. En la otra denominada SQL dinámico, una sentencia puede ser modificada total o parcialmente por el propio programa durante su ejecución, lo que proporciona una mayor flexibilidad que puede ser útil en algunos casos especiales, pero suele requerir técnicas de programación más avanzadas en el manejo dinámico de memoria y variables.

Componentes de un sistema de base de datos relacionales con SQL




**Los componentes que interactuan entre usuarios y sistemas.**

No existe ningún estándar que determine los tipos de datos que pueden ser almacenados en una base de datos relacional. *Diferentes sistemas de base de datos* relacionales de diferentes fabricantes pueden almacenar diferentes tipos de datos, aunque no existe una diferencia tan marcada entre ellos. El tamaño y formato de cada uno de los tipos de datos varían de una máquina a otra (PC, mainframe, macitoch, etc.) y de un sistema a otro (Sybase, Progress, Oracle, etc.).

# ELEMENTOS QUE COMPONEN UNA SENTENCIA SQL

El SQL es el lenguaje que se utiliza para pedir al DBMS que realice las operaciones deseadas sobre las tablas, utilizando:.

1. **Palabras predefinidas**. Son aquellas que tienen un significado predefinido por SQL. Todas las sentencias de SQL empiezan con una de estas palabras. Ejem: SELECT, UPDATE, INTO.

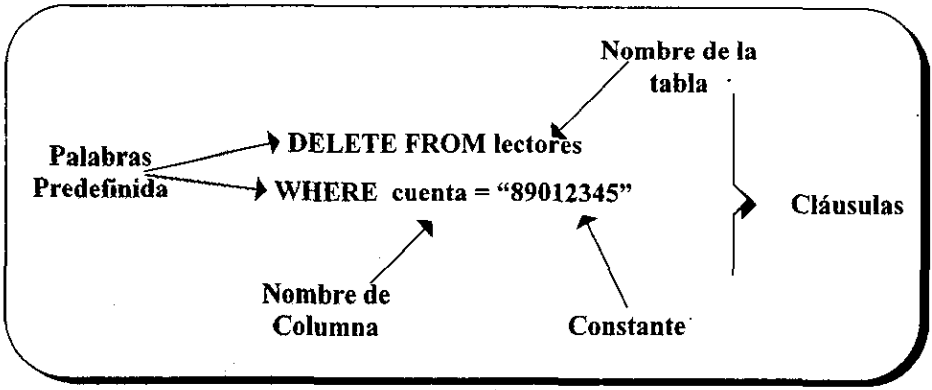
| | | | | |
|---|---|---|---|---|
| all | dec | grant | open | sql |
| and | decimal | group | option | sqlcode |
| any | declare | having | or | sum |
| asc | default | in | order | to |
| accses | desc | insert | pll | union |
| begin | double | int | precision | unique |
| between | end | integer | primary | update |
| by | escape | into | priviliges | user |
| character | exec | is | procedure | values |
| check | exists | language | public | view |
| close | fetch | like | real | whenever |
| cobol | float | max | references | where |
| commit | foreing | min | rollback | with |
| continue | fortran | module | section | work |
| count | found | not | select | |
| create | from | null | setavg | |
| current | go | of | smallint | |
| cursor | goto | on | some | |

2. **Nombre de tablas y /o columnas**. Son palabras definidas por el usuario de la base de datos. Ejem: Lectores, Catalogo, Tesis, para las tablas y nombre, cuenta, rfc, para columnas.

3. **Constantes**. Son secuencias de caracteres (letras, números, signos) que representan un valor determinado. Cuando este no es número debe ir entre apóstrofes. Ejem: titulo = "SABZ", Lim_libros = 4, Lim_dias = 6.

4. **Signos delimitadores**. Signos especiales que sirven para delimitar a los elementos anteriores dentro de la sentencia cuando no van entre comillas o apóstrofes (El más utilizado es uno o más espacios en blanco, pero también lo son los paréntesis, la coma, entre otros).

, ( ) < > . : = + - * / <> <= >=




**Estructura de una sentencia SQL**

## TIPOS DE SENTENCIAS SQL

De acuerdo con el tipo de operación que expresan las sentencias, se pueden clasificar en los siguientes tipos:

- Sentencias de manipulación de datos (***Data Manipulation Language***), permite realizar consultas y mantenimiento de datos:

a) **SELECT.** Extrae datos de una o varias tablas, se utiliza para consultas.

b) **INSERT.** Permite añadir una o varias filas a una tabla.

c) **UPDATE** Modifica uno o más valores de una o más filas de una tabla.

d) **DELETE.** Borra una o más filas de una tabla.

- Sentencias de definición de datos (***Data Definition Language***), permiten definir nuevos objetos o destruir objetos existentes, como **CREATE** define nuevas tablas y **DROP** permite destruirlas.

- Sentencias de control de datos (***Data Control Language***), controla aspectos varios como lo es la confidencialidad de acceso a los datos, con la sentencia **GRANT** permite conceder autorización a un usuario determinado para acceder a una tabla determinada, y **REVOKE** niega la autorización.

A continuación se presenta una recopilación de comandos básicos de SQL y la respectiva sintaxis de las sentencias.

## DML (Data Manipulation Language/Lenguaje de Manejo de datos LMD)

## SELECCIÓN

El concepto más fundamental del SQL se denomina bloque de consulta cuya forma básica es:

**SELECT** (lista de columnas)
**FROM** (lista de tablas)
**WHERE** (predicado)

Las columnas son pertenecientes a las tablas definidas en la cláusula FROM, si en lugar de los nombres se especifica un * significa que son todas las columnas de la tabla.

El predicado de la cláusula ***Where*** es una condición que puede ser verdadera o no. El resultado de ejecutar una sentencia ***Select*** es siempre una tabla, aunque sea de una fila con una columna.

Existen ciertas palabras claves que realizan una selección más especifica y *que se especifican dentro de la cláusula* **where**, *tales como:*

- Operadores de comparación (=, <, >, <=, >=).
- Rangos (**between** y **not between**) Define un rango de valores.
- Igualdad de caracteres (**like** y **not like**) Busca cadena y/o subcadenas.
- Valores desconocidos (**is null** y **is not null**) Valores nulos.
- Listas (**in** y **not in**) Define una lista de valores permitidos.
- Combinaciones de condiciones (**and**, **or**).

CLAUSULA ORDER BY

Por definición, los registros en una relación en la tabla carecen de un orden predefinido. Por medio de la cláusula ***"order by"*** devuelve los registros de una selección en orden ascendente o descendente.

Formato:

SELECT * o (lista de columnas)
FROM nombre de la tabla o nombre de la vista
WHERE condición de búsqueda
ORDER BY nombre de la columna ASC o DESC

CLAUSULA GROUP BY

Permite agrupar los datos de una selección basados en el contenido de una columna. Formato:

SELECT * o (lista de columnas)
FROM nombre de la tabla o nombre de la vista
WHERE condición de búsqueda
GROUP BY nombre de la columna

CLAUSULA DISTINCT

Los contenidos de los registros de datos pueden determinar los resultados de una selección. Se pueden agrupar registros distintos basados en su contenido. La palabra reservada ***distinct*** elimina los registros duplicados en los resultados de una selección. Formato:

SELECT DISTINCT lista de selección

FROM nombre de la tabla o nombre de la vista

WHERE condición de búsqueda

Existen cinco funciones agregadas en SQL. Estas funciones operan sobre las filas recuperadas de una selección y el resultado es un único valor, las funciones son:

| Función | Proceso |
|---|---|
| Maximum | Devuelve el valor máximo encontrado entre las filas seleccionadas. |
| Minimum | Devuelve el valor mínimo encontrado entre las filas seleccionadas. |
| Sum | Devuelve la suma de los valores seleccionados de la filas seleccionadas. |
| Count | Devuelve el contador de las filas seleccionadas. |
| Average | La media aritmética de las filas seleccionadas. |

**JOIN** El join recupera datos de dos o más tablas.

**CATINV**

| cveart | descrip | existencia |
|---|---|---|
| 145236 | computadora | 10 |
| 15121512 | escritorio | 15 |
| 101010 | anaquel | 60 |

**INVENTARIO**

| cveart | desdet |
|---|---|
| 101010 | 4x1.5m sin cubierta |
| 101011 | 4.5x .1m c/cubierta. |
| 101111 | silla ejecutiva |
| 145512 | 1.5x .80m c/2caj. |
| 145236 | 486 32MRAM 1GB 133MH |

**RESULTADO**

| cveart | descrip | existencia | desdet |
|---|---|---|---|
| 145236 | computadora | 10 | 486 32MRAM 1 GB 133MH |
| 15121512 | Escritorio | 15 | 1.5x .80m c/2caj. Con llave |
| 101010 | anaquel | 60 | 4x1.5m sin cubierta |

**NOTA**: Cuando ***n*** tablas son ligadas con un join, por lo menos ***n-1*** condiciones de join son necesarias para obtener el producto cartesiano.[1]

Sintaxis:

SELECT (tabla.nombre_columna, ...)
FROM (tabla1, tabla2,...)
WHERE condición de búsqueda

- En la cláusula where deberá tener por lo menos un join valido, el cual no puede ser nulo (por que nulo no puede ser igual a nulo). Las columnas dentro de la condición de join necesitan no estar en una cláusula select.

- Las columnas con el mismo nombre en múltiples tablas, deberán ser precedidas por el nombre de la tabla. Los nombres de las columnas deberán ser los mismos para las columnas del join.

## SUBQUERIES (SELECT ANIDADOS)

Un subquery es una sentencia select usada como parte de una expresión dentro de otra sentencia select, update, insert o delete. El subquery es resuelto en la respuesta de otro query en la cláusula where.

Los subqueries son usados porque algunas veces son más fáciles de comprender que un join o cuando es imposible utilizar un join.

[1] **Producto cartesiano.** Si no se especifica para que renglones es el join de diferentes tablas, el sistema asumirá que se desea el join para todos los renglones de cada tabla Llamado producto cartesiano o producto cruzado.

**Subqueries Restricciones:**

- Un subquery no puede incluir cláusulas order by o compute o palabras reservadas.
- La palabra ***distinct*** no puede ser usada con subqueries que incluyen una cláusula group by.

Un subquery puede contener una o más subqueries ejemplo:

```
SELECT  nombre
FROM proveedores
WHERE  cvepro =  (SELECT  cveprov
                  FROM ordyfac
                  WHERE  cveoryfac = (SELECT  cveoryfac
                                      FROM bono
                                      WHERE  aplicada = "T"))
```

**Nota**: Si el subquery regresa múltiples valores se deberá incluir la cláusula in, además pueden incluir operadores de comparación

## INCREMENTO DE FILAS.

La sentencia insert incrementa los registros de una tabla.

**INSERT** (nombre de la tabla) o (nombre de la vista)
**INTO** (nombre de columna1, nombre de la columna2, ...)
**VALUES** (literal, NULL) o sentencia select

Una sentencia insert que tiene una cláusula values inserta una sola fila.

**BORRADO DE FILAS**

La sentencia delete borra filas de una tabla, la sintaxis a seguir es:

**DELETE FROM** (nombre de la tabla) o (nombre de la vista)
**WHERE** (condición de búsqueda)

Si se omite la cláusula where se borrarán todas las filas de la tabla.

**CAMBIOS EN LAS FILAS.**

La sentencia update modifica la información de las filas de una tabla:

**UPDATE** (nombre de la tabla) o (nombre de la vista)
**SET** (columna = expresión) o (columna = NULL)
**WHERE** (condición de búsqueda)

La sentencia update que omita el where cambiara todos los registros de una tabla, ya sea a nulos o a un valor dado. Si y sólo si la cláusula where es verdadera, se realizarán los cambios.

## DDL (Data Definition Language/Lenguaje de Definición de datos LDD)

**VISTAS.**

Una vista crea una tabla virtual ensamblando una o más tablas existentes. Una vez que la vista se ha creado, se podrá acceder a ella como a una tabla. Con una vista, una parte seleccionada de una tabla existente aparece como una tabla separada con un único nombre. Una vista puede también traer columnas de varias tablas.

**CREATE VIEW** nombre de la vista
(nombre de la columna de la vista, ....)

*Cualquier sentencia select puede crear una vista*

**Nota**: Cada una de las sentencias SQL tiene típicamente más de una posible variante. Existen varias modalidades para cada sentencia, permitiendo combinación de estas abarcando accesos de datos más complejos y completos por las sentencias.

## CREANDO TABLAS

**CREATE TABLE** nombre de tabla
( columna1 tipo de dato [UNIQUE], columna2 tipo de dato [NOT NULL], ...)

*La identificación de la columna como única es opcional, así como la* especificación de que acepte nulos o no. Las partes opcionales de una sentencia SQL aparecen entre corchetes. Los puntos suspensivos indican repetición.

**Ejemplo:**

CREATE TABLE deposito
(n_suc CHAR(15), n_cuenta INTEGER, n_ch CHAR(20), saldo REAL);

## ELIMINANDO TABLAS

La sintaxis para la eliminación de tablas es:

**DROP TABLE** *nombre de la tabla*

Cualquier información que contenga la tabla se perderá

La sentencia DROP, no debe confundirse con borrar todos sus elementos, Por ejemplo, DELETE * FROM lectores borra todos los registros, pero la tabla lectores sigue existiendo. DROP TABLE lectores; elimina la tabla.

## INDICES

Los índices son el único elemento que se puede considerar parte de la definición del esquema físico parte del estándar SQL. Sin embargo son elementos lógicos pues ya que se pueden eliminar y añadir en cualquier momento sin afectar al funcionamiento del sistema (Sólo a su rendimiento). Esto quiere decir que las aplicaciones y consultas funcionarán exactamente igual desde el punto de vista lógico con índices que sin ellos. Cuando existen índices el DBMS puede decidir utilizarlos (o no) para acelerar el acceso.

Hay que tener en cuenta que los índices, si bien aceleran los accesos (consultas) también ocupan espacio en disco y necesitan un tiempo para actualizarse cuando hay modificaciones en la tabla.

SQL permite un índice de múltiples columnas (conocido como índice compuesto), ya que en muchos casos, el acceso a la base de datos se mejora con índices de este tipo. La sentencia para crear un índice es:

**CREATE INDEX** (nombre del índice)
**ON** partes (columnas que lo conforman)

El índice creado ayudara a encontrar más rápido los datos de la base.

## DCL (Data Control Language/Lenguaje de Control de datos LCD)

Se hace referencia a la parte del lenguaje de SQL que se ocupa de los apartados de seguridad y de la integridad en el procesamiento concurrente

## PERMISOS

Los permisos controlan el acceso a la base de datos, cada uno de los permisos concede al usuario el realizar alguna operación, los permisos y privilegios que se conceden son:

| Permisos. | Privilegio concedido. |
|---|---|
| alter | Cambiar la estructura de las tablas |
| delete | Borrar filas de una tabla |
| insert | Añadir filas a una tabla |
| select | Seleccionar filas de una tabla |
| update | Cambiar filas de una tabla |
| all | Todos los permisos |
| index | Crear o eliminar índices. |

La sentencia ***grant*** concede a un usuario permisos hacía la base de datos con la siguiente sintaxis:

**GRANT (ALL** ó **ALL PRIVILEGES )** ó
**UPDATE** ó **INSERT** ó **DELETE** ó **ALTER** ó **INDEX** ó **SELECT**
(columna1, columna2, ....)
**ON** (nombre de la tabla, nombre de la tabla2, ....)
**TO** (identificador de usuario) ó **PUBLIC**,.... [whit grant opción]

La sentencia ***GRANT ALL*** asigna todos los posibles permisos al usuario, la palabra reservada ***PUBLIC*** concede permiso a todo el mundo para acceder a la base de datos. La concesión de permiso individual es por nombre del permiso e identificador de usuario. La sentencia ***WITH GRANT OPCION*** involucra que se permite conceder permisos, pero no se aplica si un permiso tiene ya concedido el PUBLIC, si a un permiso se le ha concedido PUBLIC cada uno de los usuarios tiene ya ese permiso.

La sentencia revoke quita permisos a usuarios:

**REVOKE (ALL** ó **PRIVILEGES** ó **ALTER** ó **DELETE** ó **INDEX** ó **INSERT** ó **SELECT** ó **UPDATE)**
**ON** nombre de la tabla
**FROM** (identificador de usuario ó **PUBLIC**), ....)

Revocar un permiso que sea public, borrará el permiso de todos los usuarios, pudiendo también revocar permisos de usuarios individuales por medio de esta sentencia.

**Nota**: La sentencia revoke no forma parte del estándar ANSI, variando su sintaxis de un sistema a otro.

## ÁREA PARA COMUNICACIÓN DEL SQL

Cuando se ejecuta un programa que tiene sentencias SQL embebidas, estas son ejecutadas por el DBMS y luego este devuelve al programa los datos pedidos, si existen. Además, debe devolver una información que indique al programa si su petición se ha ejecutado correctamente o no. Para ello, se usa un área definida dentro

del programa que se llama área de Comunicación de SQL (en inglés SQLCA: SQL Communication Area).

La SQLCA se define dentro del programa, según los convenios de definición de datos propios de cada lenguaje. Es una área que debe tener una estructura preconvenida, con unos campos y formatos determinados, en los cuales el DBMS almacenara después de ejecutar cada sentencia SQL la información necesaria para indicar al programa si la ejecución fue correcta o no. En este último caso además proporcionará información para conocer la causa o circunstancia que han impedido que la ejecución fuera la correcta.

En todo módulo fuente en el que haya sentencias SQL embebidas debe haber una SQLCA definida (y no más de una).

Consta de varios campos, pero el más frecuentemente tratado en los programas es el que contiene el código de retorno SQLCODE. Este código es un número que nos permite saber si se realizó con éxito la petición. Para ello se sigue el convenio siguiente:

- **Si el código de retorno es 0.** **La sentencia se ejecutó correctamente.**
- **Si contiene un valor negativo.** **Se produjo un error.**
- **Si contiene un valor positivo.** **La sentencia se realizó bien, pero con alguna condición de excepción.**

La definición de la SQLCA se hará con las sentencias disponibles en cada lenguaje para definir estructuras con campos numéricos y alfanuméricos. La forma de codificarla será por tanto dependiente de cada lenguaje.

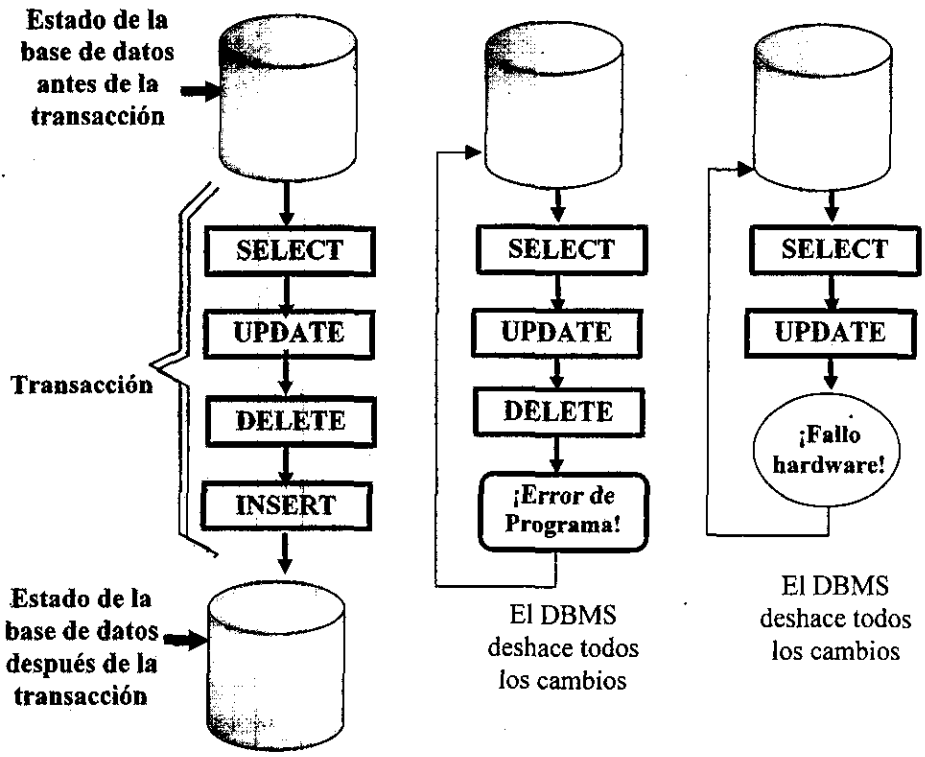
Puesto que toda sentencia de SQL incorporada ejecutable puede generar potencialmente un error, un programa bien escrito comprobará el valor de SQLCODE después de cada sentencia SQL.

## TRANSACCIONES

Una transacción es una unidad de trabajo que resulta de uno o más cambios a la base de datos. Se pueden agrupar varias interacciones o modificaciones en una transacción.

También se pueden aceptar o descartar, al final de la transacción todos los trabajos incluidos en la transacción. Las sentencias SQL son las que controlan el que se acepten o rechacen los trabajos realizados dentro de una transacción.

El mecanismo de transacción asegura que o bien todo el trabajo se complete o bien que se descarten todos los cambios.




Una transacción ayuda a asegurar que los datos introducidos en la base de datos son correctos. La transacción comienza cuando comienza la entrada de un movimiento. Al final del proceso de entrada, la transacción pasa o falla. Si la entrada de datos es incorrecta la transacción falla, y la base de datos no salva los cambios, pero si todos los datos son correctos, la transacción se ejecuta.

La transacción es por tanto la que ayuda a mantener en una base de datos su integridad y confiabilidad de los datos.

La sentencia ***commit*** y ***rollback*** controlan las transacciones, la sentencia commit aplica a una base de datos todos los trabajos que se llevan a cabo durante una transacción, y la sentencia rollback rechaza los cambios.

**Commit** (work)
**rollback** (work)

Algunos sistemas de gestión de bases de datos suministran una sentencia separada para comenzar el trabajo. Esta sentencia es por tanto la que comenzará el procesamiento de la transacción.

Por lo que se ha dicho, las sentencias SQL pueden formar parte de un programa, en cuyo caso se ejecutan cuando se ejecute éste, o pueden escribirse y ejecutarse directamente desde una terminal interactiva sin necesidad de escribir un programa para ello.

SQL gracias a su flexibilidad, potencia y a la capacidad de usarse interactivamente para realizar consultas no planificadas, es decir, no previamente incluidas en programas, abre a los usuarios finales la posibilidad de acceder directamente a los datos.

## APENDICE B

# SUN SOLARIS

El sistema operativo es un conjunto de programas que administran todas las operaciones de computo, y proporciona una interface entre el usuario y los recursos del sistema.

El sistema Solaris 2.x incluye:

- El Sistema Sun OS 5.x es un sistema operativo de 32 bits, basado en el sistema V Release 4 [SVR4] SO UNIX Los estándares de la Industria incluyen SVR4 como una familia de protocolos para las redes.
- La familia ONC de protocolos de red y servicios distribuidos.
- La aplicación gráfica OpenWindows 3.x. Interfaces Gráfica para el usuario. (Graphical User Interface GUI) en el manejador de ventanas OPEN LOOK.

- Practica funcionalidad para correr sobre procesadores SPARC, Intel 386, 486, Pentium y otras arquitecturas compatibles con DOS.
- Herramientas de escritorio de uso personal y herramientas para grupos de trabajo incluyendo correo electrónico con multimedia, manejador de archivos, herramientas de impresión, imágenes y otras.
- File Manager que provee un gráfico e intuitivo camino para navegar dentro de los sistemas de archivos locales y remotos.
- Image Tool es una herramienta para despliegue de imágenes de hasta 40 formatos distintos. Audio Tool. Graba y reproduce sonidos de mensajes hablados, ellos pueden ser conectados a un correo electrónico.
- OPEN LOOK Admintool es la base para la administración local.
- Arquitectura avanzada que incluye un completo multiprocesamiento y una sofisticada multilectura.
- Network Information Services Plus (NIS+).
- Multimedia mail. Correo electrónico con multimedia, simplifica el envío de mensajes incorporando audio, gráficas, incluyendo archivos.

El sistema Operativo UNIX (OS) esta basado en archivos, la orientación de estos habilita el software de UNIX para correr sobre diferentes plataformas – desde supercomputadoras hasta computadoras individuales -.

El sistema operativo también hace uso de memoria virtual[1] o espacio swap, y las funciones que maneja usan procesos conocidos como daemons [2](demonios).

[1] El sistema operativo **con memoria virtual** habilita aplicaciones para correr como si hubiera más memoria disponible que la que existe físicamente, hace esto usando espacio de disco duro temporalmente como memoria de almacenamiento, también es conocido como espacio swap.

[2] Los **daemons** o demonios son importantes para la funcionalidad del sistema operativo, son programas que corren en el fondo para administrar funciones de sistema tal como la impresión.

Existen tres principales partes del sistema operativo:

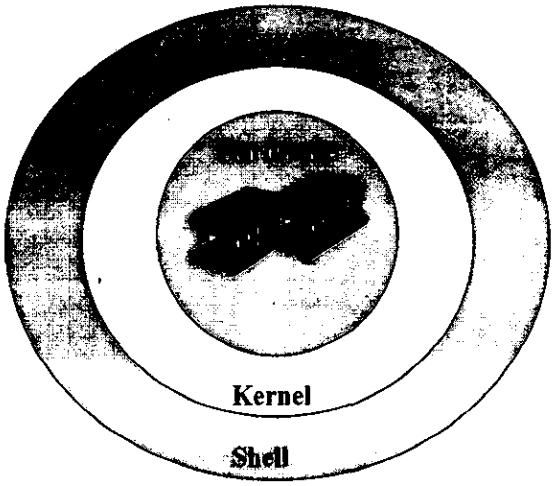
- Kernel
- Shell
- Estructura de archivos

**El Kernel.-** Es el corazón del sistema operativo. El kernel proporciona las siguientes funciones:

- Manejo de dispositivos, memoria, proceso y demonios.
- Control de funciones (transmisión de información) entre los programas del sistema (utilerías) y el hardware del sistema.
- Ejecuta todos los comandos.
- Maneja funciones tales como:
  - ✧ Espacio Swap. Parte reservada del disco por el Kernel para usar durante el procesamiento.
  - ✧ Daemons. Procesos que ejecutan tareas particulares del sistema.
  - ✧ Sistema de archivos. Una jerarquía de directorios, subdirectorios y archivos.

**El Shell.-** Es una interface entre el usuario y el kernel. Actúa como un interprete o traductor de comandos. El shell acepta comandos emitidos por el usuario, y los interpreta para enviar la ejecución al kernel. Existen tres tipos de shell disponibles en Solaris:

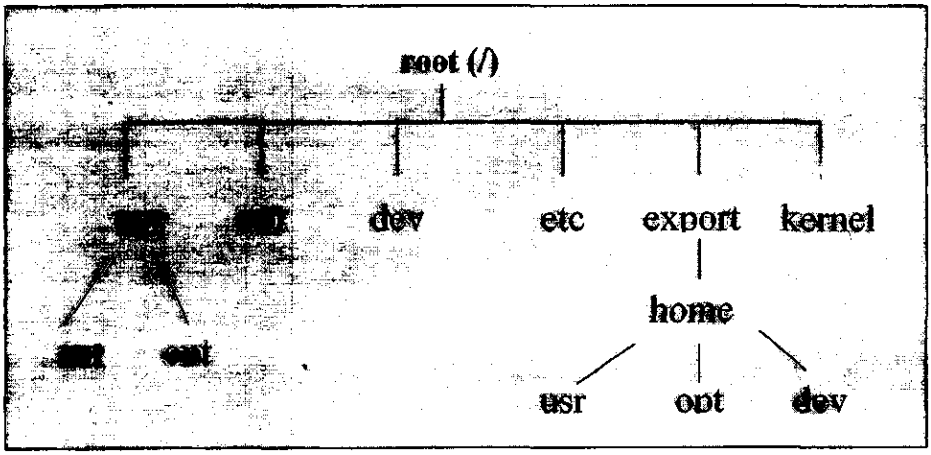
- Bournel shell ($) Es el shell por default.
- Korn shell ($) Es una mejora del anterior, tiene características adicionales como: manejo de alias, historial, edición de comandos en línea.
- C shell (%) La sintaxis es similar al lenguaje C, de ahí el nombre, tiene características similares al Korn Shell.




**La estructura de archivos.-** Tiene una jerarquía de directorios, subdirectorios y archivos que están organizados o agrupados para propósito especifico.

***Directorio.-*** Es una locación para otros archivos o subdirectorios.

***Subdirectorio.*** Cualquier directorio dentro de otro directorio.



El directorio principal es llamado root (/) y es el punto inicial del sistema de archivos jerárquico. El directorio root es requerido por la función del sistema operativo ya que contiene archivos críticos (tales como el Kernel, llamado */kernel/geniunix*).

- ***/usr*** Contiene comandos ejecutables, utilidades de administración de sistema y rutinas de librerías.

- */etc* Contiene archivos de administración, tales como el *passwd* y el archivo hosts.

- ***/opt*** Contiene tres partes del software de aplicación.

- ***/export/home*** Consta de los directorios de usuarios.

El UNIX estándar, SVR4, abarca las principales variantes de UNIX (System V, BSD, SunOS y Xenix), uniendo a la mayoría de las bases instaladas de usuarios UNIX. El ambiente de operación de Solaris, basado en SVR4, da a los diseñadores de software, administradores de sistema, y a los usuarios finales los beneficios de un sistema operativo normal incluyendo una amplia compatibilidad, un buen crecimiento y un tiempo reducido de comercialización. También proporciona un poder y funcionalidad en el producto reflejado por años de refinamiento. Entre las muchas ventajas que el ambiente del sistema operativo provee son la portabilidad, interoperabilidad, escalabilidad y compatibilidad.

## CONFIGURACIÓN DE SISTEMAS DE SUN

❒ *Clientes del Sistemas*

Desde el punto del hardware, los clientes del sistema son configurados en una de las siguientes formas: standalone, dataless, o diskless.

SISTEMAS STANDALONE.

Un sistema Standalone en red puede compartir información con otros sistemas dentro de la misma, sin que este se vea afectado por un aislamiento de la red en caso de que está falle, ya que tiene la capacidad de seguir trabajando por si solo.

El sistema Standalone puede trabajar por si solo, por que es dueño de su propio disco duro que contiene los sistemas de archivos root (/), /usr, /home y espacio de swap. Así el sistema Standalone tiene acceso local al software, espacio de memoria virtual y crear archivos para los usuarios.

El sistema Standalone debe de tener suficiente espacio de disco duro para poder soportar los cuatro sistemas de archivos necesarios.

Un sistema Standalone sin red, tiene las mismas características mostradas anteriormente con la única excepción que no se puede conectar a la red.

CLIENTES DATALESS.

Un cliente Dataless tiene almacenamiento local para su propio sistema de archivo root (/) y espacio de swap. El cliente Dataless no puede funcionar si es

aislado de la red, por que sus sistemas de archivos ejecutables (/usr) y home son montados a través de la red por medio del servidor.

CLIENTES DISKLESS.

Un cliente Diskless carece de disco duro y dependen de que un servidor le pueda proporcionar software y espacio de almacenamiento. Un cliente Diskless monta a través de un servidor remoto los sistemas de archivos root(/), /usr, y /home.

Los clientes Diskless generan un significante trafico de red, debido a su incesante necesidad de obtener el software de sistema operativo y espacio de memoria virtual a través de la red.

Un cliente Dataless demanda menos recursos del servidor y de la red que un cliente Diskless. Por que un cliente Dataless requiere menos acceso a la red, y un servidor puede soportar mucho más clientes Dataless que clientes Diskless.

Los clientes Diskless son menos caros para manejar que los sistemas Standalone. Los archivos de los usuarios y de todos los clientes Dataless también pueden ser almacenados y administrados centralmente sobre un servidor.

❒ *Servidores del Sistema*

Un servidor de sistema esta normalmente configurado como un sistema standalone con recursos que pueden ser acezados por otros sistemas de la red. Los servidores deberán tener discos locales y generalmente tiene un manejador de cinta y de CD-ROM. También pueden tener una o más impresoras conectadas.

## Sistema de Archivos en RED, Network File System (NFS).

Ofrecer simplemente los protocolos para comunicaciones con otro sistema no basta. Se necesita una aplicación que pueda sacar partido de los recursos del sistema remoto. Un sistema de archivos como NFS puede ofrecer estas posibilidades. NFS es un sistema de archivos distribuido desarrollado por SUN Microsystems que está construido sobre TCP/IP. Para el usuario, NFS significa que puede conectarse a diferentes máquinas y tener acceso en todas a los mismos archivos. No se necesitan órdenes o procedimientos adicionales para listar archivos, ver su contenido, crear nuevos archivos o copiar archivos al disco fijo local. NFS permite llevar una administración centralizada de los discos.

El sistema de archivos en red es un servicio que permite a los usuarios tener acceso a jerarquías de archivos como si estos fueran locales. Las jerarquías de archivos pueden ser sistemas de archivos completos o directorios individuales. Los sistemas que participan dentro de NFS pueden ser heterogéneos. Pueden pertenecer a diferentes compañías, utilizar diferentes sistemas operativos y estar conectados a redes de arquitecturas diferentes. Estas diferencias son transparentes para la aplicación del sistema de archivos de red.

**Los beneficios que NFS ofrece son:**

- Permite que múltiples computadoras puedan usar los mismos archivos, permitiendo que los mismos datos puedan ser acezados por todos a través de la red.

- Reduce los costos de almacenamiento por tener computadoras compartiendo aplicaciones en vez de tener espacio de disco local para cada aplicación del usuario.

- Provee consistencia de datos y rentabilidad por que todos los usuarios pueden leer el mismo conjunto de archivos.

- Realiza el montado de sistemas de archivos transparente para los usuarios.

- Realiza el acceso remoto de sistemas de archivos transparente para los usuarios.

- Soporta ambientes heterogéneos.

- Reduce las tareas de administración.

El sistema NFS realiza la localización física de los sistemas de archivos irrelevante para el usuario. El NFS puede usarse para habilitar a los usuarios para que puedan ver todos los archivos relevantes, sin considerar la localización, en vez de tener copias de los archivos mas comúnmente usados sobre distintos sistemas de archivos, el software de NFS permite tener una sola copia de este archivo en un solo lugar de disco acezando esta a través de la red. Bajo la operación de NFS, los sistemas de archivos remotos son indistinguibles sobre un sistema local.

**NFS: Servidores y Clientes.**

Los términos cliente y servidor son usados para describir los papeles que una computadora juega cuando comparte el sistema de archivos. Si un sistema de archivos reside en el disco de una computadora y esa computadora hace el sistema de archivos disponible para otras computadoras en la red, entonces esa computadora actúa como un *servidor*. La computadora que esta acezando a ese sistema de archivos se dice que es un *cliente*. El software de NFS habilita cualquier computadora dada para acezar a cualquier otra computadora a su sistema de

archivos, al mismo tiempo, proporcionar acceso a su propio sistema de archivos. Una computadora puede estar de cliente, servidor, o ambos en cualquier momento en la red.

Un servidor puede proporcionar archivos para un cliente de Diskless (un Diskless es una computadora que no tiene disco duro). Un cliente Diskless cuenta con un servidor que le provee totalmente con un conjunto de archivos para poder interactuar con la red. Un cliente Diskless puede ser solo un cliente nunca como un servidor, debido a que este no tiene un dispositivo de almacenamiento como el disco duro.

Cuando los clientes accesan a los archivos en el servidor, el sistema de archivos ha sido montado para ser compartido.

Cuando un cliente entra a un sistema de archivos remoto, este no puede hacer una copia del sistema de archivos; más bien, al entrar a un sistema de archivos remoto se crean una serie de llamadas de procedimientos remotos esto habilita al cliente para acceder al sistema de archivos y ser transparente en los servidores de disco. Al entrar a un sistema remoto parece como si entráramos a un sistema local y los usuarios teclean órdenes como si el sistema de archivos fuera local.

Una vez que se ha compartido un sistema de archivos en un servidor por una operación NFS, este puede ser acezado por los clientes.

**EXPORTAR.**

*Exportar es un proceso en el que un servidor de sistemas de archivos en red* (NFS) permite el acceso de sus archivos a clientes remotos. Pueden ser exportados tanto en directorios individuales como sistemas de archivos, pero las entidades que

se exportan se conocen como sistemas de archivos. El proceso de exportación puede realizarse de manera automática cuando la máquina se enciende, o por medio de un comando especifico que el administrador de la red ejecute una vez que la maquina ya esta funcionando.

## MONTAR.

Montar es el proceso mediante el cual un cliente NFS tiene acceso a los sistemas de archivos que fueron exportados por un servidor NFS. Cuando se exportan los sistemas de archivos o los directorios NFS, se ponen a disposición de los clientes a través de la red por una serie de llamadas de procedimiento remotas que permiten al cliente tener acceso a los archivos de manera transparente, desde el disco del servidor. Los sistemas de archivos que se montan no están fisicamente en la máquina del cliente, pero el procedimiento de montado permite que el usuario ejecute comandos como si el sistema de archivos fuera local.

## Mejoras para los administradores de Sistemas.

Para los administradores de Sistemas, Solaris ofrece una variedad de nuevas utilerias para simplificar las tareas del administrador, estas incluyen:

- **Información de Dispositivos.** Los administradores pueden usar las utilerias opcionales para obtener información acerca de los dispositivos instalados incluyendo nombres de dispositivos, atributos, y accesibilidad.

- **Administración de Sistemas de Archivos.** Estas utilidades habilitan a los administradores para crear, copiar, montar, depurar, reparar y desmontar sistemas de archivos.

- **Información del Sistema.** Estas utilerias reportan la memoria y configuración del sistema. El administrador del sistema puede usar estas utilerias para cambiar los nombres de los sistemas y el nodo de la red.

- **Manejador de procesos.** Esta utileria controla las tareas del sistema. Usando esta utileria el administrador puede generar reportes sobre el rendimiento (performance), los usuarios, accesos a disco, etc. En adición el administrador puede cambiar de nivel de sistema, matar procesos activos etc.

- **Manejador de usuarios y grupos.** Con estas utilerias el administrador de sistemas puede crear y borrar entradas a bases de datos asignando password, especificar por default, etc.

- **Contabilidad del Sistema**. La utilerias de contabilidad ayudan a los administradores a monitorear el uso del sistema, procesador, usuarios, paquetes recibidos, errores etc.

- **Admintool.** El Admintool puede correrse bajo el ambiente OpenWindows y provee facilidades para el manejo del sistema, es de gran ayuda para agregar hosts, manejo de la red y muchas otras tareas cotidianas en el sistema local.

**Particiones de Disco.**

Sobre sistemas SPARC, Solaris define ocho particiones de disco y determina a cada una un uso convencional. Estas particiones son numeradas del la 0 a la 7. A continuación se explicara el contenido de cada partición.

| PARTICION | SISTEMA DE ARCHIVOS | CLIENTE/ SERVIDOR | FUNCION |
|---|---|---|---|
| 0 | root | ambos | Contiene los archivos y procedimientos que despiertan al servidor |
| 1 | swap | ambos | Provee memoria virtual o espacio de swap. Swap es espacio usado cuando se corren programas demasiado grandes en la computadora y estos son puestos en el área de swap |
| 2 | ---------- | ambos | Por convención se refiere a el disco entero. Este es definido automáticamente por el comando format de Sun y los programas de instalación de Solaris. El tamaño de la partición no puede ser cambiado. |
| 3 | /export | solamente en el servidor | Es contenido en versiones alternativas de sistemas operativos. Estas alternativas son requeridas por los sistemas clientes, cuyas arquitecturas difieren a la del servidor. Los clientes con arquitecturas semejantes a las del servidor obtienen archivos ejecutables los sistemas de archivos /usr que se encuentra usualmente sobre la partición 6. |
| 4 | /export/swap | solamente en el servidor | Provee a los clientes espacios de memoria virtual. |
| 5 | /opt | ambos | Contiene software de aplicación que será instalado en el sistema. |
| 6 | /usr | ambos | Contiene comandos del sistema operativo conocidos como archivos ejecutables *diseñados para ser corridos por los* usuarios. Esta partición también contiene documentación, programas del sistema y librerías del sistema. |
| 7 | /home o /export/home | ambos | Contiene todos los archivos creados por los *usuarios.* |

# GLOSARIO

- **Arreglo**. Lista uni o bidimensional de variables con subíndice.

- **Base de datos**. Colección integrada de datos almacenados en un dispositivo de almacenamiento de acceso directo.

- **Ciclo**. Serie de instrucciones de programa que se ejecuta repetidamente hasta que se satisface una o más condiciones.

- **DBMS**. (Database Management system) sistema de administración de base de datos. Un conjunto de programas empleado para definir, administrar y procesar la base de datos y sus aplicaciones.

- **Depuración**. Proceso de detección y corrección de errores en programas de computadoras o sistema.

- **Diccionario de datos**. Recurso que informa acerca de las características de los datos contenidos en un sistema computacional.

- **Diagrama de flujo**. Herramienta gráfica de análisis y diseño de sistemas que hace posible plasmar por medio de esquemas visuales el flujo de los datos a través de un sistema.

- **Estación de trabajo**. Es toda aquella computadora conectada a la red.

- **FDDI** (Fiber Distributed Data Interface) Red de fibra óptica a alta velocidad, 100Mbps, basada en Token-Passing, con distancias de transmisión mayores de 2 Km.

- **Interface.** Se dice de cualquier procedimiento, tanto hardware como software que sirva de enlace y coordinador entre dos sistemas o dispositivos que emplean distinto tipo de señal.

- **Interface Gráfica de usuario**. (GUI) Una interface que posee ventanas, símbolos gráficos, menús desplegables, y otras estructuas que se manipulan con frecuencia con un apuntador de ratón.

- **Kernel.** La parte medular de un sistema operativo. El kernel proporciona servicios a todos los programas que ejecutan bajo el sistema operativo, y también impide que los usuarios y programas interfieran entre sí.

- **Modelo entidad – relación.** Los esquemas y convenciones empleados para crear un modelo de los datos de los usuarios. Las cosas en el mundo de los usuarios se representan con entidades, y las asociaciones entre estas cosas se representan con relaciones. Normalmente los resultados se documentan en un diagrama entidad – relación.

- **Nodo.** Cada elemento conectado a la red por medio de una interface de red, pudiendo ser: terminales de comunicación, servidor, estaciones de trabajo, impresoras, así como también los elementos de unión de las distintas ramas de la red.

- **Programa**. Conjunto de instrucciones que ocasiona que el sistema realice acciones especificas

- **Protocolo.** Conjunto de normas y reglas que permiten establecer una comunicación

- **RAM** (Memoria de Acceso Aleatorio) Almacena temporalmente programas y datos que se están utilizando en el momento. Con la falta de energía se borra.

- **RED (Network).** Conjunto de computadoras interconectadas entre sí y/o con otros equipos cuya configuración permita que esto sea un medio para *transmitir, compartir, recibir y procesar información y recursos.*

- **Ruteador o Router.** sistema utilizado para transferir datos entre redes que utilizan el mismo protocolo. Un router puede ser un dispositivo software o hardware o una combinación de ambos.

- **Servidor de Base de Datos**. En una red con arquitectura de base de datos Cliente/Servidor, la computadora que ejecuta el DBMS y procesa acciones sobre la base de datos en nombre de sus computadoras clientes.

- **Sistema Cliente/Servidor**. Un sistema de dos ó más computadoras en el cual al menos una de ellas proporciona servicios para otra o para las demás. Los servicios pueden ser de base de datos, de comunicaciones, de perifericos o de algunas otras funciones.

- **Sistema Operativo**. Es una colección de programas que coordinan las actividades del sistema computacional, teniendo como misión general: controlar las operaciones, tales como asignación de recursos, planeación de recursos y trabajos, así como el monitoreo de actividades.

- **Sistema Operativo de Red**. Es decir, el software base de control bajo el que se trabaja, tal como: NetWare, LAN Manager, OS/2, LANtastic y Appletalk. Este sistema reside en el servidor y cada estación de trabajo cuanta con un componente de software que permite que una aplicación sea leída y se pueda escribir datos en el servidor desde la máquina local que se esté utilizando.

- **Sistema de Software**. Bajo términos estrictos de computación es un conjunto u ordenación de elementos organizados para llevar a cabo algún método, procedimiento o control mediante el procesamiento de información.

- **Software**. Los programas de computadora, las estructuras de datos y la documentación asociada, que sirven para realizar el método lógico, procedimiento o control requerido.

- **SQL.** ( Structured Query Language). Lenguaje de consulta estructurada. Un lenguaje para definir la estructura y procesamiento de una base de datos relacional. Se emplea como un lenguaje de consulta único o puede incorporarse en programas de aplicación. El American National Standards Institute (ANSI), es el encargado de reglamentarlo.

- **Subclase.** *Una clase más abajo en la jerarquía de herencia que su padre, la* superclase. Cuando crea una nueva clase, con frecuencia se le llama hacer una subclase.

- **Superclase.** Una clase más arriba en la jerarquía de herencia que su hijo, la subclase.

- **T1.** Terminología Bell que se refiere a un sistema de portadora digital *usada para la transmisión de datos a través de la jerarquía telefónica.* Velocidad de transmisión de 1.544 Mbps.

- **Topología Física.** Se refiere a la forma física en la que se conectan y distribuyen las computadoras. Las tres topologías más utilizadas comúnmente son de tres tipos: bus (conexión lineal), anillo y estrella.

- **Topología Lógica.** Define la forma en que se lleva a cabo el acceso de *comunicación que puede ser una topología como Ethernet o Token Ring* y que tiene que ver precisamente con la distribución física de la red donde cada dispositivo se comunicara por medio de adaptadores de red.

# BIBLIOGRAFIA:


- Benavides Abajo, J. M. Olaizola, Bartolomé. E. Rivero, Cornelio.
  "SQL", Para Usuarios y Programadores.
  Madrid, España; 1991 Ed. Paraninfo, S.A. 383 pp.
- Cárdenas, Alfonso F.
  "Sistemas de Administración de Bancos de Datos".
  México, D. F., 1993 Ed. LIMUSA. 3ª Reimpresión 611 pp.
- Ceballos, Francisco Javier
  "Microsoft, Visual C++, Aplicaciones para Windows"
  *México, D.F. 1997 Ed. Alfa Omega, grupo editorial* S.A. de C.V. p 30.
- G. Burch, John and Grudnitski, Gary
  "Diseño de Sistemas de Información"
  México, D.F. 1993 Ed. Limusa, S.A. de C.V. 985 pp.
- Garbus, Solomon & Tretter
  "Sybase DBA survival Guide".
  USA. 1995 Ed. SAMS Publishing 508 pp.
- Gibbs, Mark; Brown, Todd Traducción: De la Barrera U., Ricardo
  "Redes para Todos"
  México, 1995. Ed. Prentice-Hall Hispanoamericana, S.A. 2ª ed. 472 pp.
- Halvorson, Michael. Traducido: Fabregat Carrascosa, Jesus
  "Visual Basic 4 para windows 95, Paso a Paso"
  México, D. F., 1995. Ed. McGraw-Hill. 372 pp.
- "Introducción a las Redes LAN de Microcomputadoras". (Apuntes)
  México, D.F. Agosto, 1996. UNAM. DECFI,
- Kendall y Kendall
  "Sistems Analysis and Design"
  New Jersey, USA 1995. Ed. Prentice-Hall. 3ª Ed. 894 pp.
- Kroenke, David M. Traducido: Martínez Sarmiento, Miguel A.
  "Procesamiento de Bases de Datos, Fundamentos, Diseño e Instrumentación"
  México, D. F., 1995 Ed. Prentice-Hall Hispanoamericana, S. A. 5ª Ed.
  605 pp.
- Lemay, Laura; L. Perkins, Charles. Traducido: Díaz Mena Jorge I.
  "Aprendiendo JAVA en 21 días"
  México, D. F., 1996 Ed. Prentice-Hall Hispanoamericana, S. A. 525 pp.
- Mahler, Paul
  "Power Builder v. 4 Desarrollo de aplicaciones Cliente/Servidor".
  México, D.F.; 1996 Ed. Prentice/Hall 410 pp.

- Moncayo Lopez, Luis Miguel
  "CONEFESZ"
  México, D.F., 1996 UNAM 313 pp.
- Moncayo López, Luis Miguel.
  "FESNET"
  México, D. F., 1994. UNAM 177 pp.
- Nelson, Ross Traducido: Suarez S., Joaquín María.
  "Guía completa de Visual Basic para Windows"
  México, D. F., 1995. Ed. McGraw-Hill 2ª ed. 395 pp.
- Novell Netware 386
  "Manual de Referencia"
  México, 1992. Ed. McGraw-Hill, Inc. 747 pp.
- Ortega Tinoco, Raquel Maribel
  "RUTEADOR, Tecnología, Comunicación y Eficiencia"
  México, D. F., 1994. DECFI, UNAM 140 pp.
- Person, Alex.
  "Cliente/server architecture"
  Serie: On Computer Conmunicatión
  USA, 1993. Ed. McGraw-Hill. 2ª ed. 569 pp.
- Pressman, Roger S.
  "Ingeniería de Software"
  México, D. F., 1993. Ed. McGraw-Hill. 3ª ed. 824 pp.
- PowerSoft
  "Mastering DataWindows, Student Guide"
  USA, 1996. Sybase, Inc.
- PowerSoft
  "Fast Track to Power Builder, Student Guide"
  USA, 1996. Sybase, Inc.
- Russel, Charlie; Crawford, Sharon.
  "Running Microsoft Windows NT Server ver. 4.0"
  España, Madrid. 1997 Ed. McGraw-Hill Interamericana, S. A. 591 pp.
- "Redes LAN de Microcomputadoras". (Apuntes)
  México, D.F. Agosto, 1996. UNAM. DECFI,
- Steven Heyman, Mark Traducido: Ruiz Faudon, Luis María.
  "La Esencia de Visual Basic 4"
  México, D. F., 1996. Ed. Prentice-Hall Hispanoamericana, S. A. 457pp.
- Stoltz, Kevin. Traductor: Sergio Luis M. Ruiz
  "Todo acerca de Redes de computación"
  México, D. F., 1995. Ed. Prentice-Hall.
- SunSoft
  "Solaris 2.4 Introduction"
  A sun Microsystem, Inc. Busines, USA 1994
- SunSoft
  "Solaris Openwindows User guide version 3.1"
  Ed. Prentice-Hall A sun Microsystem, Inc. Busines, USA 1993 309pp.

- SunSoft
  "Solaris 2.X system administration Essentias"
  A sun Microsystem, Inc. Education Services USA 1995
- Sybase
  "SQL Server Basic Operations Vol 1 Student Guide"
  USA, 1993 Server Publications Group
- Sybase
  "SYBASE SQL Server Installation guide for SunOS Release 5.x (SVR4)
  USA, 1994 Server Publications Group
- Sybase
  "Manual de Referencia de SQL Server de Sybase, ver 11.0.x
  Volumen 1: Comandos, Funciones y temas"
  USA, 1995 Server Publications Group
- Wiederhold, Gio Traductor : Ma. de Lourdes Fournier G.
  "Diseño de Bases de Datos".
  México, D.F. 1993 Ed. McGraw-Hill 2ª ed. 921 pp.
- Whitten, Bentley & Barlow
  "Systems Analysis & Design Methods".
  USA 1989. Ed. Irwin 2ª Ed. 793 pp.

**Direcciones de Internet:**

www.espol.edu.ec
www.sybase.com
www.sun.com
www.uni.texas.edu
www.diamante.uc3m.edu