9

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE ESTUDIOS SUPERIORES
CUAUTITLAN

"BASES DE DATOS: MEJORAMIENTO CONTINUO DE LA BASE DE DATOS PARA EL OTORGAMIENTO DE CREDITOS DE UNA INSTITUCION BANCARIA"

TRABAJO DE SEMINARIO

QUE PARA OBTENER EL TITULO DE

LICENCIADO EN INFORMATICA

P R E S E N T A

ALBERTO MANZANO CORDOVA

ASESOR:
ING. VICTOR HUGO ARROYO HERNANDEZ

CUAUTITLAN IZCALLI, EDO. DE MEX. 2000

FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLAN

UNIDAD DE LA ADMINISTRACION ESCOLAR

DEPARTAMENTO DE EXAMENES PROFESIONALES

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

DEPARTAMENTO DE EXÁMENES PROFESIONALES

DR. JUAN ANTONIO MONTARAZ CRESPO
DIRECTOR DE LA FES CUAUTITLAN
P R E S E N T E

ATN: Q. Ma. del Carmen García Mijares
Jefe del Departamento de Exámenes
Profesionales de la FES Cuautitlán

Con base en el art. 51 del Reglamento de Exámenes Profesionales de la FES-Cuautitlán, nos permitimos comunicar a usted que revisamos el Trabajo de Seminario:

Bases de Datos: Mejoramiento Continuo de la Base de Datos para el Otrogamiento de Créditos de una Institución Bancaria

que presenta el pasante: Alberto Manzano Cordova

con número de cuenta: 9131395-3 para obtener el título de

Licenciado en Informática

Considerando que dicho trabajo reúne los requisitos necesarios para ser discutido en el EXÁMEN PROFESIONAL correspondiente, otorgamos nuestro VISTO BUENO.

A T E N T A M E N T E
"POR MI RAZA HABLARA EL ESPIRITU"
Cuautitlán Izcalli, Méx. a 24 de Octubre de 2000

MODULO	PROFESOR	FIRMA
I	Ing. Víctor Hugo Arroyo Hernández	
II	MC. Araceli Nivon Zaghi	
IV	Lic. Carlos Pineda Muñoz	

A mis padres...

Por el gran esfuerzo que han hecho y por toda la sabiduría que me han dado, así como su apoyo incondicional para hacer realidad mis metas y lograr lo que soy ahora.

INDICE

CAPITULO 4. CASO PRACTICO

I N T R O D U C C I Ó N

Las bases de datos actualmente han tenido un gran avance, en cuanto al manejo de estas tanto para los usuarios expertos como los usuarios finales, ya que permite tener un mejor control de la información dentro de una organización.

El uso de las redes de computadoras que se ha dado hoy en día para el entorno de una organización, permite que la información de una base de datos sea utilizada por todas las áreas de la organización, dentro de las diversas arquitecturas que se pueden utilizar en una red esta la arquitectura cliente/servidor.

En la arquitectura cliente/servidor, el servidor de la base de datos contiene los archivos compartidos junto con las reglas de integridad de información y validación, y también algunos tipos de procesos inteligentes. Es decir la aplicación del cliente contiene los menús, los modelos, las definiciones de los reportes y el código de programa asociado con la interfaz del usuario.

El servidor realiza el procesamiento de la búsqueda de la forma que los clientes lo especifican y envía la información que resulta de las búsquedas que se hacen en la red y que van hacia el cliente. La arquitectura cliente/servidor en una base de datos es más confiable y rápida para el tráfico de información de mucho volumen, pero esto produce un costo mayor. Pero las bases de datos bajo un ambiente cliente/servidor son la mejor opción para las aplicaciones de misión crítica.

Aunque actualmente existen muchos modelos de datos, aquí se hace hincapié en el modelo relacional. El término *por relación* tiene una definición teórica muy específica, pero el uso diario describe un sistema compuesto de tablas separadas que juntas forman una base de datos.

Las tablas separadas a menudo están en una relación "de una a muchas", es decir, los registros detallados a una tabla se almacenan en otra. De esta forma se puede ver que para un negocio pequeño, puede que tenga una tabla de clientes y una de órdenes que contengan una información detallada sobre cada cliente. Las dos tablas se enlazan por un campo común, este podria ser el número del cliente.

La normalización, es otro proceso de organizar las tablas de la base de datos de forma que haya muy poca duplicidad de información como sea posible, entre todas las tablas en la base de datos, es algo crucial para lograr el modelo de relación. Es fácil normalizar una pequeña base de datos que contenga sólo un par de tablas, pero las cosas se pueden complicarse mucho a medida que la base de datos crece.

La forma en que debe de normalizarse una base de datos depende del significado que se le dé a los elementos de información. Este proceso no puede ser automático, y se requiere de un administrador experto para que la base de datos funcione de forma correcta.

Otro de los puntos importantes para una base de datos con múltiples tablas relacionadas, es mantener la integridad de referencia, es decir la constancia interna que garantizara que ningún registro se refiere a otro ya no existente. Para cada registro hijo, debe haber un registro padre.

Otro de los elementos importantes para poder obtener la información de una base de datos, es poder contar con un lenguaje que permita realizar las consultas de información. Para este manejo y manipulación de la información se utiliza el lenguaje SQL, el cual generalmente se provee con los productos de una base de datos.

Todos estos elementos y otros permiten tener un sistema de base de datos potente, flexible y fácil de usar, tanto para los usuarios informáticos como para los usuarios finales.

OBJETIVO GENERAL:

Describir los beneficios que se ofrecen al contar con una base de datos para el otorgamiento de créditos en una Institución Bancaria.

OBJETIVOS PARTICULARES:

Contar con un registro de la llamadas realizadas por los distribuidores y ejecutivos. Así como llevar el registro en cantidad de las llamadas hechas al cliente potencial, indicando su estatus y resultado.

Optimización de la generación de cotizaciones y del otorgamiento de autorizaciones a los clientes de los diferentes planes de crédito y de los plazos.

CAPITULO 1
GENERALIDADES

1.1 ANTECEDENTES DE LAS BASES DE DATOS.

Una de las partes más importantes dentro de una organización es él poder contar con información legible y fácil de obtener. En un principio el procesamiento de datos dentro de una organización era a través de cintas de papel y en tarjetas perforadoras. Los equipos de computo de esa época podían leer las cintas y las tarjetas. Este proceso consistía en agrupar los datos de tal forma que se generaban registros físicos y una vez que se tenían objetos del mismo tipo que se repetían en cierto número de veces, se creaba una colección para constituir un fichero.

Los reportes o informes se generaban mediante el uso de programas de aplicación, los cuales procesaban los datos para obtener la información, usando compiladores para traducir los programas de aplicación del lenguaje fuente a instrucciones de máquina.

El tiempo que se tardaba en recuperar los datos la computadora de las cintas magnéticas, muchas veces era muy lento, sobretodo cuando sé tenía un gran numero de información, lo cual ocasionaba un gran costo para las empresas en ese tiempo. La creación de los discos magnéticos permitió un acceso más rápido a los datos, así como la forma de guardar la información. De esta forma se fue creando el concepto de Bases de Datos.

En un principio las bases de datos fueron conocidas como *Sistemas de información de gestión*. Uno de los primeros sistemas de bases de datos fueron creados por los lenguajes de programación existentes en esa época tales como: COBOL, BASIC, etc., los cuales estaban a cargo de los programadores. Estos sistemas eran muy complejos y difícilmente eran fáciles de comprender, a esto también hay que agregarle el mantenimiento de dichos sistemas, lo que ocasionaba un gran costo para las empresas de aquella época.

El uso de estos del *Sistema de Información de Gestión* no tuvo mucha fuerza dentro de las organizaciones, ya que en un principio la forma en que se desarrollo este software no fue lo suficientemente sofisticado, como para responder a las necesidades de las organizaciones.

A principios de la década de los setenta se da a conocer el concepto de bases de datos. En el año de 1963 en un simposio en Santa Mónica (Estados Unidos) se da a conocer el concepto *Data Base*. En el año de 1967 el grupo de estandarización Codasyl introdujo la expresión de *Data Base Task Group*.

El grupo Codasyl (Conference on Data System Langueges) estaba representado por un grupo de personas privadas, así como de miembros del gobierno de Estados Unidos. Este grupo se dedicaba al estudio y normalización de bases de datos principalmente en el lenguaje de programación COBOL. Lo que pretendia el grupo era resolver los problemas de estandarización de los sistemas de gestión.

Con el surgimiento del concepto de base de datos surgen los Sistemas de Gestión de Bases de Datos (SGBD), que son los que controlan las bases de datos. Los SGBD permiten un mejor control sobre la base datos, ya que estos son los que permiten el acceso a la información, mantienen la integridad de los datos, es decir es la interfaz que permite conectar al usuario para que pueda manejar y manipular la información que se encuentra en una base de datos.

Los manejadores de bases de datos como Clipper, Dbase, Foxpro, etc., son sistemas para usuarios finales, los cuales son fáciles de utilizar pero no son muy potentes dentro de una gran organización. Debido a la demanda de información surgen sistemas más potentes los cuales tienen muchas ventajas como: la independencia de datos, accesos múltiples, recuperación de información rápida y fácil, etc . dentro de estos sistemas se pueden mencionar Oracle, Sybase, SQL Server, etc.

En la actualidad el mercado de las bases de datos es muy grande, ya que existe mucha competencia de las empresas dedicadas al desarrollo de estos productos

Actualmente se han desarrollado grandes mejorías. tal es el caso de las bases de datos para usuarios finales. las cuales han permitido que los usuarios de poca experiencia puedan crear sus propias bases de datos sin tantas complicaciones, ya que la mayoría de estos productos tienen una manera rápida y fácil de crear bases de datos.

1.2 BENEFICIOS DE LAS BASES DE DATOS.

Los beneficios que han aportado las bases de datos a las organizaciones han sido muchos de los cuales se puede mencionar el tener un mejor control centralizado de los diversos datos que se manejan dentro de dichas organizaciones, ya que esta información como ya se sabe es uno de los activos más valiosos que puede tener una organización.

El poder contar con un buen control de datos en una organización tiene muchos beneficios dentro de estos se pueden mencionar los siguientes:

1. Independencia de los datos

   El contar en con una independencia de los datos en un sistema de base de datos proporciona una gran ventaja dentro de una organización, ya que esto permite hacer cambios y aplicaciones sin que se modifique, tanto el diseño lógico como el físico de la base de datos. De esta forma se permite que se introduzca nueva información, que se elimines datos, así como cambios en la forma en que sé accesa, etc., sin que se tengan que modificar los programas o interfaces para los usuarios finales.

   La independencia de los datos proporciona una gran ventaja ya que evita muchas veces la reprogramación de los programas que permiten al usuario accesar a la información cuando se producen cambios en los datos.

   La facilidad que se proporciona con la independencia de los datos, evita muchas veces que las organizaciones reduzcan sus costos al tratar de adaptar los sistemas de información con los constantes crecimientos de las organizaciones.

Aunque la independencia de los datos muchas veces no es alcanzada totalmente. los sistemas de hoy en dia permiten una mayor independencia de los datos permitiendo a los usuarios una mayor flexibilidad en los datos y en la elaboración de estos.

2. Coherencia de los Resultados

El contar con un buen diseño para una base de datos permite que la información que se tiene sea coherente y perfectamente entendible para los usuarios finales Facilitando de esta forma la obtención de la información.

El tener coherencia en los datos elimina la redundancia de estos permitiendo desaparecer los problemas al tener que actualizar datos, ya que esto anteriormente obligaba a actualizar una serie de ficheros. Con esto también se elimina inconveniente de las divergencias en los resultados debidas a actualizaciones no simultaneas en todos los ficheros.

3. Reducción de espacio de almacenamiento.

Los sistemas de bases de datos actuales han proporcionado grandes beneficios en cuanto al almacenamiento de datos, ya que con las nuevas técnicas de almacenamiento y de compactación de datos han permitido una menor ocupación de almacenamiento.

Otro de los factores importantes en la reducción de datos es el desarrollo de nuevos componentes de almacenamiento que permiten almacenar grandes cantidades de datos. Dentro de estos componentes se pueden mencionar los CD, los ZIP, las cintas, entre otros.

4. Acceso rápido y fácil para los usuarios finales.

Los usuarios lo que pretenden de una base de datos son contar con un rápido acceso a la información. Actualmente los sistemas de bases de datos permiten una gran facilidad para la obtención de la información concentrada en estas. Con el perfeccionamiento y la creación de nuevos algoritmos de búsqueda se ha mejorado el acceso a las bases de datos.

La facilidad que permiten las bases de datos de hoy en día, al poder accesar a través de diferentes plataformas ha permitido que la gente de sistemas puedan desarrollar interfaces más fáciles de programar y de esta forma permiten a los usuarios finales una mayor disposición y accesibilidad a la información de una organización.

5. Facilidad para compartir datos.

   El que una organización pueda contar con una base de datos bien diseñada permite a los usuarios tener una mejor disponibilidad de los datos para todos aquellos que tienen necesidad de estos, siempre y cuando los usuarios estén autorizados para su acceso.

   Esto también permite desarrollar nuevas aplicaciones que operen con los mismos datos almacenados dentro de la base de datos. Es decir, los nuevos requerimientos para las nuevas aplicaciones pueden solucionarse sin tener que crear nuevas bases de datos.

6. Seguridad al acceso de datos.

   Al tener normas de seguridad sobre los datos se puede asegurar que el único medio para accesar a la base de datos es a través de las normas establecidas por el administrador de datos, así como de la propia organización. De esta forma se restringe el acceso, ya sea para recuperación, modificación, alta o borrado de datos.

La utilización de un sistema de base de datos puede otorgar un gran servicio en una organización, dando de esta forma información coherente y útil para la toma de decisiones. Es por esta razón que se deben de tomar todas las medidas necesarias para poder obtener todas las ventajas posibles de los sistemas de base de datos y de esta forma poder tener un buen rendimiento de la información existente, así como de los grandes costos que se originan al implementar un sistema de base de datos en una organización.

1.3 DESVENTAJAS DE LAS BASES DE DATOS

Como todos las sistemas, las bases de datos no sólo presentan ventajas, sino que también presentan ciertas desventajas. Algunas de las desventajas o inconvenientes que pueden tener las bases de datos son:

1. Instalaciones costosas.

   La instalación de una base de datos en una organización puede acarrear un costo muy elevado, tanto en la compra del hardware como del software. Esto se debe a que muchas veces se debe de comprar y crear nuevas instalaciones o ampliar las instalaciones.

   En cuanto al software también se requiere comprar nuevos sistemas operativos dependiendo de las plataformas que se manejen, herramientas para la programación, compiladores, etc.

2. Personal capacitado.

   La administración, el diseño, la programación y la instalación de una base de datos requiere de personal capacitado. El cual pueda afrontar cualquier tipo de situación que se presente.

   El contar con un buen equipo de personas de sistemas, puede garantizar el desempeño de una base de datos eficiente, aunque esto muchas veces requiere de costos elevados.

3. Implementación larga y difícil de aplicar.

   Debido a la complejidad muchas veces de los sistemas de información, esto provoca que muchas veces la implementación de una base de datos se desfase del periodo previsto. Esto puede provocar que el instalar un sistema de base de datos sea en ciertas ocasiones una tarea larga y laboriosa.

Como se puede observar el no contar con un buen personal de sistemas, así como de un buen diseño de la base datos y de un buen plan de trabajo puede ocasionar que las grandes ventajas que proporcionan los sistemas de bases de datos, sean más bien inconvenientes y problemas para la organización.

1.4 CONCEPTOS BASICOS DE BASE DE DATOS

Para poder tener una mejor apreciación de lo que es una base de datos en este apartado se presentan los términos más usados en los sistemas de bases de datos.

Entidad

Una entidad puede ser cualquier objeto de interés referente a un dato que puede ser coleccionado. Este objeto debe distinguirse de otros objetos. Por ejemplo personas, lugares y objetos que describen entidades. Las entidades en una base de datos representan a las tablas.

Atributos

Los atributos son cualidades adicionales o identificadores de una entidad. Los atributos son las columnas o campos en una tabla. Un ejemplo de atributos de una entidad. Tenemos una entidad llamada Empleado sus atributos serian el nombre, el apellido paterno y materno, edad, etc.

Relaciones

Una entidad puede tener una relación con otra entidad. Las relaciones son definidas por llaves primarias y llaves secundarias, las cuales mantienen la integridad referencial. Las relaciones pueden señalarse en las columnas de las tablas, esto también depende del tipo de relación, ya que existen tres tipos de relaciones.

Relación Uno - a - Uno

Una entidad en A está relacionada a lo mucho con una entidad en B y la entidad en B a su vez está relacionada a lo mucho con una entidad en A. En la Fig. 1.1 se puede apreciar de una forma más dinámica.

Fig. 1.1 Relación Uno-Uno

Relación Uno - a - Muchos

Una entidad en A está relacionada con un número cualquiera de entidades en B. Una entidad en B. sin embargo puede estar relacionada a lo mucho con una entidad en A. En la Fig.1.2 se puede ver una representación gráfica.

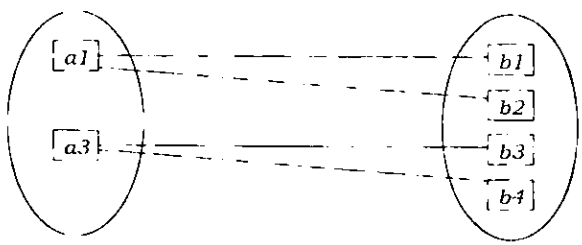


Fig. 1.2 Relación Uno-Muchos

Relación Muchos - a - Uno

Una entidad en A está relacionada a lo mucho con una entidad en B. Una entidad en B, sin embargo, puede estar relacionada con un número cualquiera de entidades en A. En la Fig. 1.3 se puede apreciar la relación.

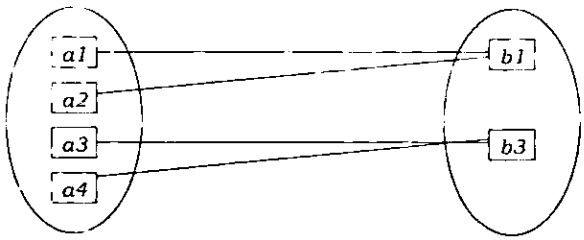


Fig. 1.3 Relación Uno-Muchos

Relación Muchos - a - Uno

Una entidad en A está relacionada con un número cualquiera de entidades en B y una entidad en B está relacionada con un número cualquiera de entidades en A. En la Fig. 1.4 se puede ver esta relación.

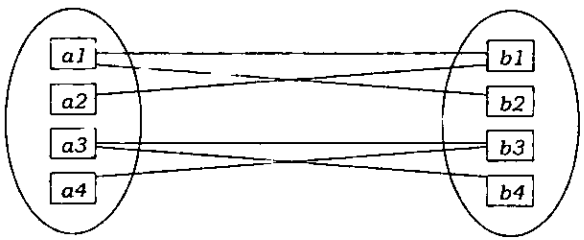


Fig. 1.4 Relación Muchos-Uno

Llave Primaria

La llave primaria es la columna o grupo de columnas que permiten la integridad. Algunas restricciones de las llaves primarias son:

- Todas las entidades de una base de datos deben de tener una llave primaria.
- Solo puede haber una llave primaria por entidad.
- En una llave primaria no se permiten valores nulos o duplicados.

Llave Secundaria

Una llave secundaria es aquella que hace referencia a una llave primaria en una entidad. Esta puede estar dentro de la misma entidad. Una columna puede tener ambas llaves. Las llaves secundarias al contrario de las primarias, estas si permiten valores nulos.

CAPITULO 2
BASES DE DATOS

2.1 DEFINICION DE BASE DE DATOS.

Con el paso del tiempo la definición de base de datos ha ido cambiando por ejemplo muchos autores dicen que es un conjunto de archivos, otros en su lugar de este termino, utilizan los términos de datos, registros, entidades, etc. Este cambio se da porque en la actualidad una base de datos ya no esta compuesta de varios archivos o registros, como era anteriormente. Pero al analizar las diferentes definiciones se puede ver que muchos autores coinciden en ciertos elementos, pero también es cierto que en muchas de estas definiciones se carecen de elementos fundamentales.

Como se puede ver a continuación algunas definiciones de diversos autores coinciden en ciertos elementos.

"Conjunto de datos de la empresa memorizado por un ordenador, que es utilizado por numerosas personas y cuya organización esta regida por un modelo de datos."

"Una base de datos es una colección de datos interrelacionados. Es decir, es la colección compieta de datos, punteros, tablas, índices, diccionarios, etc.

"Una base de datos es un conjunto estructurado de datos registrados sobre soportes a los que el ordenador puede acceder para satisfacer a varios usuarios.

Como se puede ver en las definiciones anteriores todos los autores están de acuerdo en que una base de datos es una colección o conjunto de datos, los cuales se procesan a través de una computadora. Por esta razón se puede dar la siguiente definición, la cual en cierta forma integra todos los elementos que conforman una Base de Datos.

Base de Datos

Una base de datos es una colección de datos registrados en entidades y otros objetos que se encuentran organizados y que se presentan para un servicio especifico a los que una computadora puede accesar para satisfacer simultáneamente a varios usuarios de forma selectiva y en un tiempo adecuado.

2.2 ESTRUCTURA DE LAS BASES DE DATOS.

La estructura de una base de datos es una de las partes más importante de esta, ya que en esta parte se analizan todos los elementos que conforman una base de datos para una organización. La estructura de una base de datos se compone principalmente de tres niveles los cuales son: el Nivel Interno, el Nivel Conceptual y el Nivel Externo Fig 2.1, estos niveles siempre van a depender uno del otro, generalmente la parte que se analiza primero es la del nivel interno.

La estructura de una base de datos bien definida y analizada, garantiza el buen funcionamiento y satisfacción de los usuarios dentro de una organización, así como su justificación y costo que implica la creación e instalación de un sistema de base de datos.

Nivel Interno

El nivel interno de una base de datos es la parte fisica, en esta parte es donde se especifican que tipo de dispositivos periféricos son los que se van a utilizar para poder almacenar la información que genera la organización.

En la actualidad con los grandes avances en tecnologia sé han podido crear nuevos dispositivos de almacenamiento que permiten a los responsables de la administración de la base de datos un mejor aprovechamiento de todos los recursos de espacio requeridos, para el almacenamiento de la información generada en la base de datos.

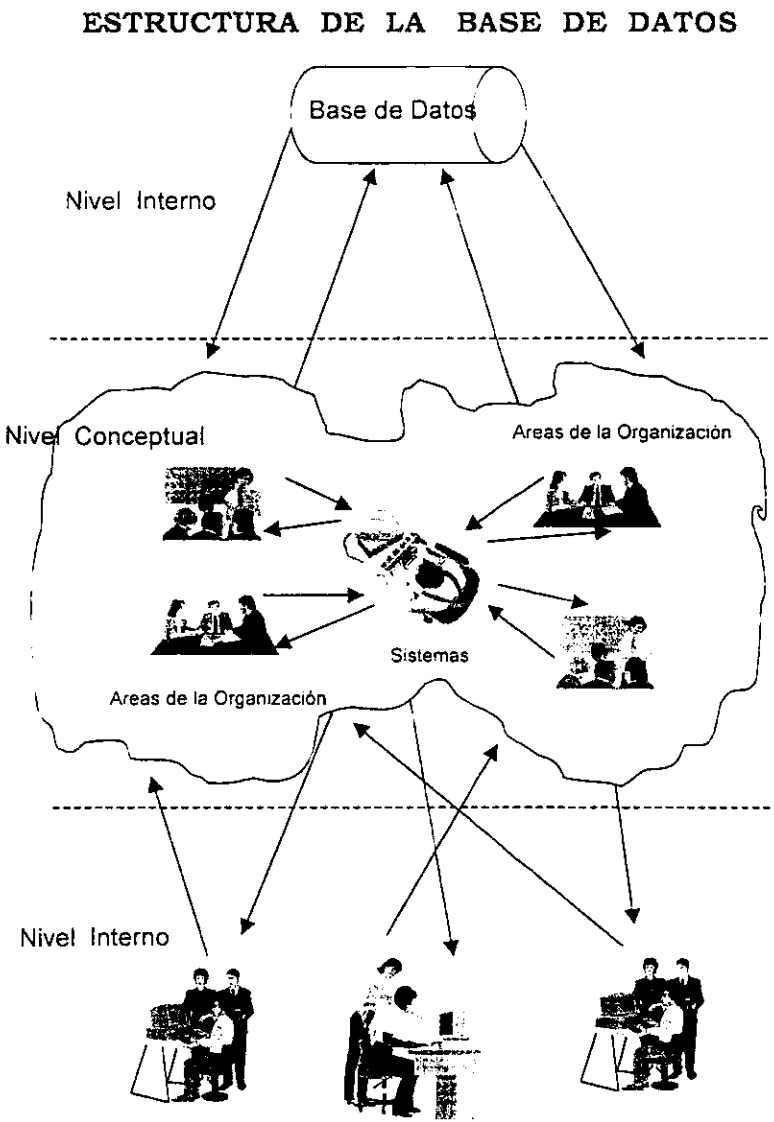


Fig. 2 1 Estructura de la Base de Datos

Por otra parte, el perfeccionamiento que se le han dado a los algoritmos de compresión de datos permite un mayor aprovechamiento de espacio tanto en disco como en cualquier otro medio de almacenamiento.

En este nivel como ya se menciono anteriormente, es donde se decide cual es el tipo de periférico más recomendable para guardar una base de datos. En esta parte también se analiza cuanto espacio va a ocupar una base de datos en el servidor o en cualquier otro tipo de estación de trabajo.

En el nivel interno también se especifica los tipos de acceso que se van a utilizar para la recuperación de los datos, así como las técnicas de clasificación de estos. Otro punto que se debe de analizar es el respaldo de los datos, es decir, en que componentes se respaldaran los datos - cintas, disco o cd's -, también se debe de especificar si estos respaldos deben ser diarios, por semana o mensuales. Esto dependerá del tamaño de la información que se genere en la organización. En la Fig. 2.2 se pueden ver los elementos que interactuan en el nivel interno

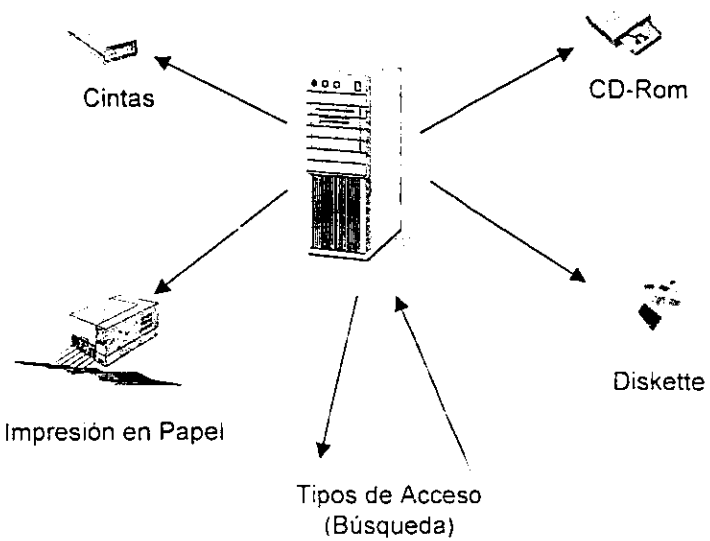


Fig 2.2 Nivel Interno

Nivel Conceptual

El nivel conceptual es la parte más importante de la estructura de un sistema de base de datos. Se puede decir que es la parte esencial, por esta razón es muy importante que su diseño se analice cuidadosamente. De este análisis depende el buen desempeño de una base de datos en una organización.

En este nivel es donde participan los elementos de una organización, ya que la función principal del nivel conceptual es describir en términos abstractos la realidad de una organización, así como los procesos de las diferentes áreas de la organización que van a participar para la creación de la base de datos.

El diseño se lleva a través de un proceso llamado modelo de datos y este modelo se determina de acuerdo a las necesidades de la organización. Existen actualmente varios modelos de datos, pero la modelización consiste principalmente en clasificar los diferente elementos de la organización y de esta forma asignarles un determinado nombre.

Los sistemas de base de datos de hoy en dia nos proporcionan un lenguaje de definición de datos que permite establecer de una forma más accesible y fácil la parte conceptual. Aunque esto es de gran ayuda, siempre se debe de analizar de manera minuciosa con los usuarios, los elementos que van a formar parte de la base de datos.

Nivel Externo

El nivel externo es la parte final, es decir es la forma de presentar a los usuarios finales la parte del nivel conceptual. Para esto se deben de desarrollar programas de utilización que permitan a los usuarios ver de una forma agradable la información que ellos necesiten.

El desarrollo de estos programas generalmente recibe el nombre de mascaras El diseño de estas mascaras debe ser de una forma tal, que no puedan ser afectadas por los cambios en el almacenamiento físico o en el método de acceso a la base de datos, así como en los cambios que se vayan dando con el tiempo en el nivel conceptual.

En este nivel sólo se deben de presentar a los usuarios aquellos datos que requiera de una forma gráfica y accesible, de tal forma que ellos puedan comprender la información que proporciona la base de datos.

Otro de los puntos que también se debe de tomar en cuenta en el análisis de este nivel es la seguridad de los datos, es decir que tipo de atributos va a tener cada usuarios. Esto debido a que la base de datos no sólo va a funcionar para una área de la organización, sino que la utilizaran dos o más áreas y cada una de estas generalmente tendrá diferentes formas de trabajar.

Es por esta razón que se deberá de especificar que usuarios podrán modificar, guardar o consultar la base de datos. Para que de esta forma se tenga un control más exacto de este.

2.3 SISTEMA DE GESTION DE BASE DATOS

La parte principal de una base de datos es la de simplificar y facilitar el acceso a los datos de una organización. El buen funcionamiento va depender de la eficiencia de las estructuras de datos utilizadas para representar dichos datos en la base de datos y de la capacidad de operar sobre estas estructuras de datos que el sistema tiene.

El garantizar el buen funcionamiento en una base de datos, es un factor importante para la satisfacción de los usuarios. Lo que permite que los usuarios, interactuen con la base de datos es el Sistema de Gestión de Datos (SGBD).

El Sistema de Gestión de Datos es *Un conjunto de programas, funciones, procedimientos y lenguajes que permiten a los usuarios las herramientas necesarias para describir, recuperar y manipular una base de datos.*

El sistema lo que permite, es sobre todo organizar los datos, así como el proporcionar los procedimientos de búsqueda y de selección de dichos datos. Aunque estas no son las únicas funciones que proporciona el SGBD. A continuación se describen las funciones más importantes de los SGBD.

Función de Descripción

El SGBD debe permitir tanto al administrador de la base de datos como a los usuarios describir el conjunto de datos que se almacenaran, la estructura de estos datos y las relaciones que puedan existir entre ellos. El SGBD debe también permitir la integridad de los datos, los permisos de acceso, las reglas de seguridad, así como las caracteristicas de tipo físico y las vistas lógicas de los usuarios.

El SGBD realiza la función de descripción a través de su lenguaje de descripción o definición de datos, cabe mencionar que este tipo de lenguaje es diferente en cada

SGBD, pero todos los SGBD deben de proporcionar los medios para definir la estructura de los datos (niveles de descripción) y de esta forma especificar las características de los datos a cada uno de estos niveles

Los niveles de descripción se pueden dividir en dos tipos los cuales son:

- Descripción Física
- Descripción Lógica

En la descripción física el SGBD debe de indicar el espacio requerido para la base de datos, las características de los campos como son; su longitud, su modo de representación (binario, decimal, alfanumérico, flotante, etc.). A parte de esto debe de permitir definir los tipos de acceso.

En la descripción lógica el SGBD debe proporcionar las herramientas para la definición de las entidades y su identificación, los atributos de cada una de estas, la relación que pueda existir entre cada una de estas entidades, los permisos de acceso a los usuarios, las restricciones de integridad, la seguridad, etc.

Función de Manipulación

La función de manipulación en un SGBD es la que permite hacer tanto a la gente de sistemas como a la de otra área, búsquedas, añadir, suprimir o hacer modificaciones de acuerdo a las reglas y normas establecidas por el encargado de la base de datos.

El lenguaje de manipulación de datos es la medio que permite realizar la función de manipulación, esta herramienta facilita la realización de estas tareas. La forma como se realizan estas tareas es a través de un conjunto de instrucciones, es decir un lenguaje propio del SGBD, o a través de lenguajes de programación que se pueden comunicar al SGBD para obtener los datos requeridos por los usuarios.

Los SGBD actualmente tienen sus propios lenguajes, aunque hoy en día los

SGBD han diseñado lenguajes que ayudan principalmente a los usuarios finales y de esta se les facilite a estos, el poder manipular los datos que se encuentran dentro de una base de datos.

Función de Utilización

El SGBD con su función de utilización ofrece al usuario la posibilidad de interactuar de una forma más accesible y a través del diálogo, con el fin de buscar, seleccionar y modificar los datos.

Como ya se sabe dentro de una organización existen dos tipos de usuarios los informáticos y los no informáticos. Dependiendo de esto es como se va a especificar el tipo de diálogo en el SGBD.

Los usuarios de mayor experiencia que son los informáticos, generalmente van a tener un diálogo mediante procedimientos de algoritmos. Mientras los usuarios no informáticos tendrán un diálogo ya definido de antemano, es decir reglas que le permitan realizar las tareas que este requiera para manejar la base de datos.

Las funciones mencionadas anteriormente son parte de las características de un SGBD en la Fig. 2.5 se puede ver la forma como interactuan estas funciones en el SGBD.

Otras de las características importantes en un SGBD son las que se menciona a continuación:

Integridad

Con el aumento de información almacenada en una base de datos, puede ser mayor el riesgo de que los datos almacenados no reflejen la realidad. El SGBD permite al usuario definir reglas que mantengan la integridad de la base de datos.

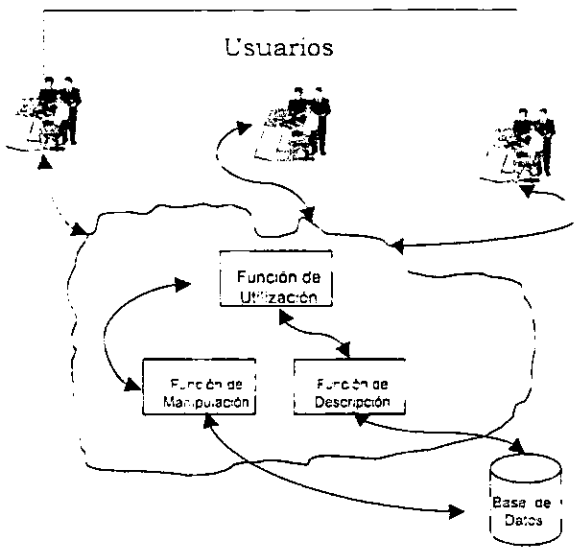


Fig 2.5 Funciones del SGBD.

Las reglas que permiten realizar esta tarea reciben el nombre de restricciones de integridad. Las cuales son propiedades que deberán cumplirse en la base de datos, no importando los valores almacenados.

La integridad permite tener una base de datos confiable para la organización, ya que con las restricciones de integridad la base de datos no permitirá que se almacene información que no cumpla con dichas restricciones y de esta forma se garantiza tener un buen control de la base de datos.

Un aspecto importante de las reglas de integridad es que se pueden almacenar en el diccionario, como una parte integral de la descripción de los datos, lo cual proporciona ciertas ventajas tales como:

- Las reglas de integridad son más fáciles de entender y de cambiar, facilitando de esta forma su mantenimiento.

- Con esto se puede detectar la inconsistencia.

- Se protege mejor la integridad, ya que ningún usuario podrá realizar programas que violen las reglas de integridad.

Confidencialidad

La confidencialidad se refiere a la seguridad de la base de datos, ya que por lo general son varios usuarios los que comparten una misma base de datos y solamente algunas personas tendrán autorización para entrar a cierta información restringida para los demás.

El SGBD proporciona mecanismos que permiten verificar y asignar los tipos de acceso a los usuarios para poder accesar a la información que este almacenada en la base de datos. Estos permisos se van a otorgar de acuerdo al perfil del usuario y estos pueden ser:

- Usuario con derecho a crear, borrar o modificar objetos y que aparte de esto puedan conceder privilegios a otros usuarios sobre los objetos que este haya creado.

- Usuario con derecho a consultar la base de datos o para realizar actualizaciones, pero sin permisos para crear, modificar o borrar objetos.

Los SGBD generalmente deben permitir realizar las siguientes funciones:

- Mantienen registrados los objetos a los que puede accesar un usuario y las operaciones que realice sobre estos objetos.
- Permiten al administrador de la base de datos definir las reglas para los permisos.
- Mantienen las identificaciones de los usuarios, así como sus contraseñas.

De esta forma el SGBD proporciona la seguridad a la base de datos de accesos no autorizados

Concurrencia

Debido a que el acceso de los usuarios a una base de datos en una organización se realiza al mismo tiempo, esto puede ocasionar conflictos de acceso y que posiblemente no se conserve la consistencia de los datos.

El SGBD debe de contar con herramientas que permitan la detección de aquellos casos en los que se produciría un conflicto de acceso. El SGBD para controlar la concurrencia maneja las técnicas de control de concurrencia, las cuales se dividen en pesimistas y optimistas.

Las técnicas pesimistas son aquellas que establecen controles previos para evitar que se produzcan situaciones que puedan provocar inconsistencia, una de las técnicas más utilizadas son la bloqueo y las marcas de tiempo.

Las técnicas optimistas son aquellas que desde un principio suponen que no tiene porque haber problemas y si estos surgieran, estas actúan a fin de volver la base de datos en un estado consistente.

Copia de seguridad y recuperación

Como se sabe todos los sistemas informáticos, están sujetos a fallas. Las causas de fallas generalmente se dan debido a problemas con el disco duro, con el suministro de energía o problemas con los programas de aplicación.

Las fallas de esto puede ocasionar que se pierda información en la base de datos, el SGBD tiene la responsabilidad de detectar este tipo de errores y de recuperar la información o estabilizar el sistema de base de datos.

La forma como realiza la recuperación de datos el SGBD es a través de procedimientos de copias de seguridad. El SGBD posee herramientas para realizar respaldos, los cuales se pueden programar para que se hagan todos los días o por un determinado periodo.

Existen varias características que componen un SGBD, las mencionadas anteriormente son las que principalmente debe de proporcionar un SGBD, pero actualmente estos sistemas poseen muchas más, las cuales proporcionan grandes ventajas tanto para los usuarios informáticos como para los usuarios finales.

Diccionario de Datos

En una base de datos unos de los objetivos principales como ya se menciono, es que muchos usuarios compartan datos. Otro objetivo importante es proporcionar datos correctos a los usuarios. Para que la base de datos pueda cumplir con estos objetos debe de tener datos correctos, redundancia mínima y control del uso de los datos, por esta razón es importante contar con un mecanismo de control central.

La herramienta que proporcionan los SGBD para controlar y manipular la información sobre los datos en las diferentes fases de diseño, implantación, operación y expansión de una base de datos es el *Diccionario de Datos*.

El diccionario de datos es una herramienta que almacena información sobre los datos relativos al origen de éstos, descripción, relación con otros datos, uso, responsabilidad y formato. Se puede decir que el diccionario de datos es una guía y contiene el mapa de la ruta hacia la base de datos en lugar de datos de la base.

Los beneficios de usar un diccionario de datos están relacionados con la recopilación, especificación y manejo efectivo de los recursos totales de una organización. Un diccionario de datos debe de ayudar al usuario de una base de datos principalmente a:

- Comunicación con otros usuarios.
- Control de los campos de datos de manera sencilla y efectiva, es decir introducir nuevos campos en los sistemas o cambiar las descripciones de los campos.
- Reducción de la redundancia e inconsistencia de los datos.
- Determinar el impacto que pudiera haber por los cambios en los campos de datos sobre la base de datos.
- Centralizar el control de los campos de datos, como ayuda en el diseño y en la expansión del diseño de la base de datos.

Los dos objetivos básicos de un diccionario de datos son la administración y el control de los datos como un recurso, en un lugar central, a través de las fases de diseño, realización y operación, así como el establecimiento de una comunicación efectiva entre todos los que estén interesados en la base de datos.

El diccionario de datos también debe de apoyar las siguientes características:

Recursos de recuperación y de informes. Estos recursos hacen que el diccionario sea más efectivo. Los recursos deben contener requisitos tanto formales como adecuados. Los informes pueden incluir las siguientes partes:

- Lista de los campos en secuencia alfabética o en cualquier otro orden.
- Listados de referencia cruzada entre los campos de datos, grupos de campos de datos, programas que lo estan usando y las áreas que son responsables del mantenimiento de los datos.
- Descripciones de datos para los programas en el lenguaje huésped y los enfoques lógicos para los modelos externos que contienen a los programas de aplicación

Apoyo de programas de servicio. Debido a que el diccionario de datos es la parte central del medio de la base de datos, este debe ser capaz de apoyar a los diferentes programas de edición, actualización y reestructuración de las bases de datos, generadores de informes, etc.

Información de control de acceso. El diccionario de datos puede contener información tomando en cuenta el control de acceso, especificando quién puede tener acceso a qué parte de la base de datos y en que forma lo puede realizar.

Generación de programas y código de descripción de datos. El almacenar la información sobre los campos de datos, sus representaciones físicas, sus relaciones con los otros campos y sobre los modelos lógico y externo deben capacitar a un diccionario de datos para que este genere el código de descripción de datos, así como también algunos módulos de programa general tal como los módulos de acceso o entrada/salida.

Consistencia. Como ya se menciono la información contenida en un diccionario de datos debe ser completa, con un formato correcto y con las referencias cruzadas apropiadas. Para que se cumplan estos requisitos se debe de revisar la consistencia de cualquier tipo de entrada al diccionario de datos. La revisión también debe basarse en la información convertida entre los modelos conceptual, lógico, externo e interno almacenados en el diccionario de datos.

En el siguiente cuadro se muestran las características con las que debe de contar un diccionario de datos.

Diccionario de Datos

1. El diccionario de datos debe apoyar a los modelos conceptual, lógico, interno y externo.

2. El diccionario de datos debe estar integrado en el Sistema Gestor de Base de Datos.

3. El diccionario de datos debe de contener varias versiones de la documentación. Es decir:

   - Versiones de ambiente de pruebas.
   - Versiones de ambiente de producción.

4. El diccionario de datos debe presentar apoyo a la transferencia eficiente de información al SGBD.

5. Un sistema de diccionario de datos debe iniciar la reorganización de la versión de producción de la base de datos como resultado de los cambios a la descripción de la base de datos.

2.4.1 Tipos de Lenguajes del SGBD

El SGBD utiliza diferentes tipos de lenguajes y procedimientos para que pueda cumplir con sus funciones de comunicación con la base de datos. Estos lenguajes pueden estar orientados hacia la función, ya sea de definición o de manipulación y los otros tipos de lenguajes son los que están orientados a los diferentes tipos de usuarios o de procesos.

Al hablar de lenguajes de función se refiere a dos tipos de lenguajes que el SGBD maneja los cuales son:

- Lenguaje de Definición de Datos.

- Lenguaje de Manipulación de Datos.

El hablar de lenguajes de tipo de usuario se refiere a lenguajes que están dirigidos principalmente a dos tipos de usuarios, los informáticos y los no informáticos - usuarios finales -. Estos usuarios pueden tener diferentes procesos y generalmente los programadores son los que se encargan de desarrollar estos sistemas, que pueden ser sometidos a un tratamiento por lotes, con periodicidad fija o a un tratamiento interactivo. es decir realizar consultas a la base de datos.

El SGBD debe de admitir varios lenguajes de tipo anfitrión para que el usuario pueda manipular los datos. Los usuarios informáticos, como el administrador de la base de datos, los analistas, los programadores, etc. , requieren los medios potentes y flexibles mediante los cuales puedan definir, manipular y extraer los datos de la base en el lenguaje de programación que mejor se acople a sus necesidades. Aunque la estructura y sintaxis de todos estos tipos de lenguajes va a depender de cada SGBD.

Lenguajes de Definición de Datos

El lenguaje de definición de datos es aquel que permite al administrador de la base definir los datos con facilidad y precisión, especificando sus distintas estructuras.

El lenguaje de definición de datos permite al administrador de la base de datos asignar nombres a los campos, a los agregados de los datos, a los registros, etc. estableciendo sus longitudes y caracteristicas, así como las relaciones entre estos elementos; también permite especificar los identificadores, e indicar las restricciones semánticas que se han de aplicar a los diferentes objetos escritos - entidades, atributos. relaciones -.

Lenguaje de Manipulación de Datos

La manipulación de una base de datos se refiere a:

- La recuperación de información almacenada en la base de datos.
- La inserción de información nueva en la base de datos.
- La supresión de información de la base de datos.
- La modificación de datos almacenados dentro de la base de datos.

El objetivo del lenguaje de manipulación principalmente es proporcionar una interacción eficiente entre las personas y el sistema.

Un lenguaje de manipulación de datos es un lenguaje que capacita al usuario a acceder o manipular datos según estén organizados por el modelo de datos adecuado. Existen básicamente dos tipos de lenguaje de manipulación:

- Los Procedimentales.
- Los no Procedimentales.

En el lenguaje de manipulación procedimental se requiere que el usuario especifique qué datos se necesitan y como obtenerlos. Mientras que en los no procedimentales se requiere que el usuario especifique qué datos se necesitan sin especificar cómo obtenerlos.

Los lenguajes de manipulación no procedimentales son más fáciles de aprender y usar que los procedimentales. Aunque estos tienen una desventaja, debido a que el usuario no tiene que especificar cómo conseguir los datos, estos lenguajes pueden generar código que no sea tan eficiente como el que se produce por el tipo de lenguajes procedimentales.

Los lenguajes de manipulación no procedimentales están dirigidos básicamente a los usuarios finales y que requieren que se lleven a cabo ciertos procesos que muchas

veces ya están definidos o que llevan ciertas reglas para ejecutarlos

Otro tipo de lenguaje de manipulación es el llamado navegacional que recupera o actualiza los datos, aquí el programador debe de indicar el camino que debe de recorrer, a través de la estructura definida en la base de datos.

En la actualidad, la mayoría de los SGBD disponen de lenguajes de manipulación muy eficiente que permiten una mejor manipulación de los datos y de forma más accesible para el usuario final tanto para la consulta y la actualización de la información de la base de datos.

2.5 MODELO DE DATOS

Como ya se había mencionado anteriormente para poder construir la estructura de una base de datos se necesita de un modelo de datos. El modelo de datos se utiliza principalmente cuando se analiza el nivel conceptual en la estructura de una base de datos. El SGBD utiliza el modelo de datos para definir la estructura fundamental de los mismos. Actualmente todos los SGBD utilizan un modelo de datos para la creación de una base de datos.

El modelo de datos va depender del SGBD y también de la organización, ya que dependiendo de esta y de sus necesidades es como se especificara que modelo de datos le conviene mas a la organización, para que tenga un mejor desempeño de la base de datos dentro de esta.

Un modelo de datos es una colección de herramientas conceptuales para describir datos, relaciones entre ellos, semántica asociada a los datos y restricciones de consistencia.

Actualmente existen diversos modelos de datos los cuales presentan características propias y otras muy similares. Cada modelo de datos tiene sus propias elementos, así como sus herramientas y diagramas para representarse.

Los modelos de datos más importantes son:

- Modelo Entidad-Relación.
- Modelo Jerárquico.
- Modelo Orientado a Objetos.
- Modelo Relacional.

Modelo Entidad-Relación

El modelo Entidad-Relación los introdujo Peter P. Chen en el año de 1976. Según Chen este modelo puede ser usado como una base para una vista unificada de los datos.

El modelo E-R, como su nombre lo indica, está centrado en dos conceptos fundamentales el de entidad y el de interrelación. Cuando se utiliza este modelo los diseñadores deben de manejar estos términos de entidad y relación para describir a la organización.

Una entidad es un objeto que se distingue de otros objetos por medio de un conjunto específico de atributos. Por ejemplo, los atributos como nombre, edad, dirección describen un registro de un empleado de una organización. Mientras que una relación es una asociación entre varias entidades. Un ejemplo de esto es la relación que podría existir entre un proveedor y el numero de productos que este pueda tener.

Al conjunto de todas las entidades del mismo tipo y relaciones del mismo tipo se denomina conjunto de entidades y conjunto de relaciones, respectivamente.

El modelo E-R además de las entidades y relaciones, también representa ciertas restricciones a las que deben ajustarse los contenidos de una base de datos. Una de las restricciones más importantes es el de la cardinalidad de asignación, que se refiere al número de entidades a las que puede asociarse otra entidad mediante un conjunto de relación.

La estructura lógica de una base de datos puede representarse a través de un diagrama E-R como se puede ver en la Fig. 2 6

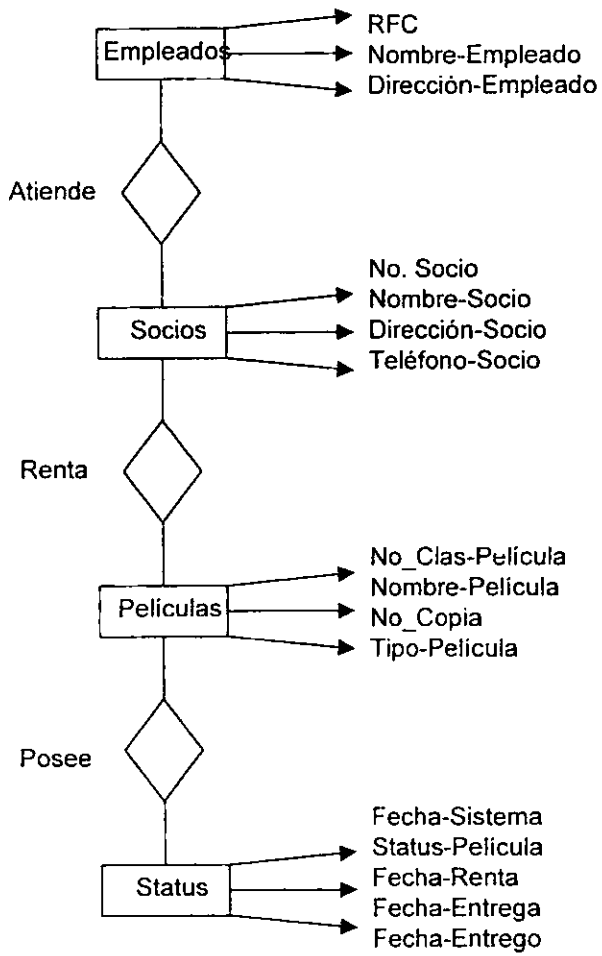


Fig. 2.6 Diagrama Entidad-Relación

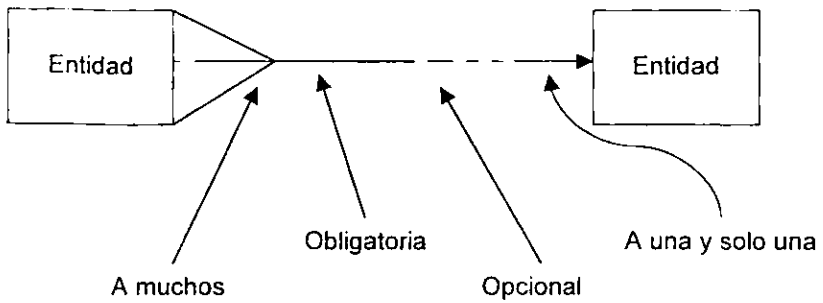


Fig 2.6 Diagrama Entidad-Relación

Modelo Jerárquico

El modelo jerárquico es un modelo que organiza a los datos en una estructura jerárquica de árbol. Una estructura jerárquica de árbol está constituida de nodos o ramas.

Un nodo es una colección de atributos de datos que describen a la entidad en ese nodo. El nodo más alto de una estructura jerárquica de árbol se llama nodo raíz. Los nodos dependientes se encuentran en los niveles inferiores.

Los nodos van a ser las clases de objetos y los arcos que unen dos nodos son las asociaciones entre dichas clases.

En un modelo jerárquico los nodos que se encuentran en el segundo nivel se conocen como hijos del nodo que se encuentra en el primer nivel y el nodo del primer nivel se conoce como el padre de los nodos del segundo nivel y así sucesivamente.

La estructura jerárquica de árbol debe de satisfacer las siguientes condiciones:

- Un modelo jerárquico siempre debe de comenzar con un nodo raíz
- Cada nodo consiste de uno o más atributos que describen a las entidades en ese nodo
- Los nodos dependientes pueden aparecer en dos niveles consecutivos. Es decir, el nodo en el nivel precedente se convierte en el nodo padre de los nuevos nodos dependientes. Estos nodos dependientes se pueden añadir tanto horizontal como verticalmente sin ningún tipo de limitaciones.
- Cada nodo que se presenta en el nivel dos tiene que conectarse con uno y sólo un nodo que se presente en el nivel uno. Cada nodo que se presente en el nivel tres tiene que conectarse con uno y sólo un nodo que se presenta en el nivel dos y esto es sucesivamente.
- Un nodo padre puede tener uno o varios nodos hijos bajo su dependencia. Cuando un nodo no tiene ningún nodo bajo su dependencia entonces este nodo no es un nodo padre.
- Cada nodo, a excepción del nodo raíz tiene que accesarse a través de su nodo padre. Es decir, la trayectoria de acceso dentro de este modelo es única.
- Un nodo puede aparecer varias veces en cada nivel. Es decir, cada vez que aparece un nodo se tiene que conectar con un nodo padre, no puede haber un nodo sin conexión, a excepción del nodo raíz.

Modelo Orientado a Objetos

Este modelo se basa en una colección de objetos. Donde un objeto contiene valores almacenados en variables instancia dentro del objeto. La diferencia con los modelos orientados a registros, es que en este modelo sus valores son objetos por si mismos.

Como en la programación orientada a objetos este modelo también maneja los términos de objeto, instancia, clase, métodos, etc.

Los métodos son las partes de código que operan sobre el objeto. Los objetos que contienen los mismos tipos de valores y los mismos métodos se agrupan en clases.

Las clases pueden ser vistas como una definición de tipo para objetos. La forma como se combinan los datos y el código en una definición de tipo es parecida al concepto de tipos de datos abstractos en lenguajes de programación

La forma en la que un objeto puede acceder a los datos de otro objeto es invocando a un método de ese otro objeto. A esto se le da el nombre de envío de un mensaje al objeto.

A diferencia de las entidades en el modelo E-R, cada objeto tiene su propia identidad única independiente de los valores que contiene. De esta forma dos objetos que contienen los mismos valores son, sin embargo distintos. La distinción entre objetos individuales se mantiene en el nivel físico por medio de identificadores de objeto.

Modelo Relacional

El modelo relacional representa los datos y las relaciones entre los datos mediante una colección de tablas, cada una de las cuales tiene un número de columnas con nombres únicos.

Este modelo se basa en la noción matemática de relación, siendo ésta un conjunto de n-tuplas, en donde se conoce n de antemano.

Una de las ventajas de este modelo es la simplicidad, ya que la facilidad de compresión por parte del usuario final. Los usuarios finales no tienen que preocupar por la estructura de almacenamiento físico.

El modelo relacional es uno de los modelos más utilizados debido a tres razones:

- La interfaz tabular la pueden entender con facilidad los usuarios tanto finales como los usuarios informáticos y además sirve como herramienta de comunicación entre ellos.
- Es sencillo y fácil convertirlo en una implementación en un sistema de información.

- El modelo relacional proporciona un criterio formal para una buena representación de datos.

Los lenguajes que se pueden usar para accesar datos en una base de datos relacional se caracterizan por su interfaz sintáctica y su poder de selección. El poder de selección de los lenguajes relacionales se basa en la completa relación, que se apoya en las expresiones del cálculo relacional.

Un lenguaje es relacionalmente completo si se puede deducir cualquier dato que se obtenga de una expresión del cálculo relacional. Uno de estos lenguajes, es el álgebra relacional, con la cual se pueden crear nuevas relaciones con base en las existentes mediante operaciones de álgebra relacional. En la Fig. 2.7 se puede ver gráficamente los elementos que componen al modelo relacional.

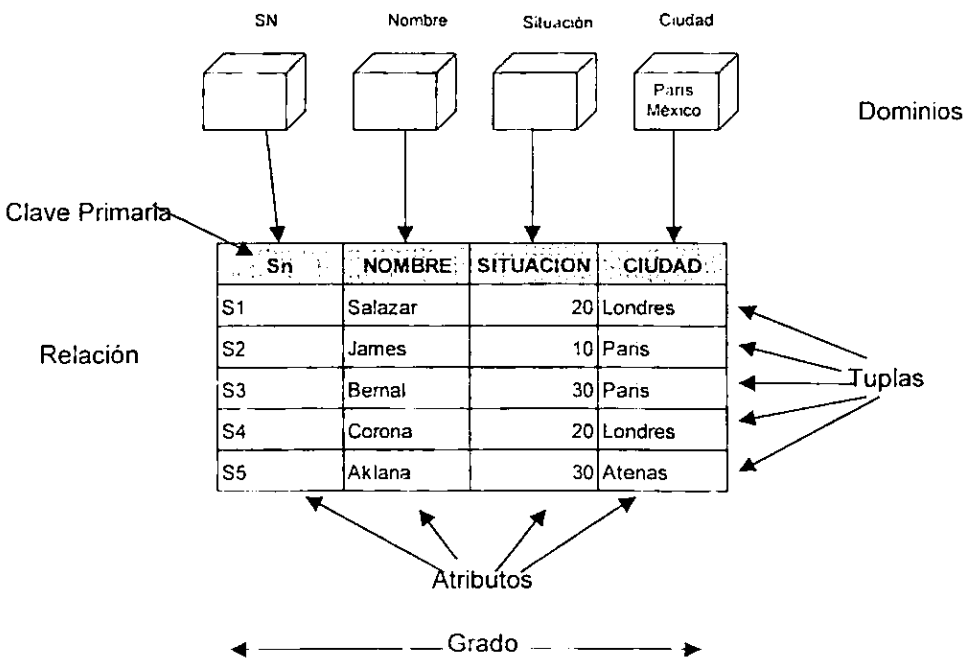


Fig. 2.7 Modelo Relacional

CAPITULO 3
MODELO RELACIONAL

3.1 DEFINICION DEL MODELO RELACIONAL

El modelo relacional se introduce como estructura fundamental de un SGBD a finales de 1970 por el Dr. E. F. Codd. En este modelo se hizo énfasis en la independencia de la representación relacional, de la implementación física en la computadora como un ordenamiento en dispositivos físicos, al indexar y usar trayectorias de acceso. El Dr. Codd también propuso en este modelo el criterio para estructurar en forma correcta las bases de datos relacionales y un lenguaje independiente de la implementación para poder operar en estas las bases de datos.

El modelo relacional representa los datos y las relaciones entre los datos mediante una colección de tablas, cada una de las cuales tiene un número de columnas con nombres únicos. Donde el modelo de datos de la organización se especifica en esta serie de tablas o relaciones, como se puede ver en la Fig. 3.1. Una especificación completa de la organización, por lo general, consiste en varias relaciones.

Los objetivos que persigue este modelo son los siguientes:

Simplicidad. Con el uso de tablas para representar las relaciones y los elementos de una organización se cumple este objetivo, ya que al mostrar al usuario final una presentación tabular, es para este más fácil de comprender los datos.

Flexibilidad. Este modelo nos presenta una mayor flexibilidad, ya que permite mostrar al usuario final, los datos de la forma en que éste prefiera y de una forma más sencilla.

Independencia Lógica. Esto se refiere, que al agregar, eliminar o modificar objetos de la base de datos no repercuta en los programas y/o usuarios que estén accediendo a

subconjuntos parciales de los mismos. Es decir, con este modelo se tiene un mayor control sobre las vistas que se proporcionan a los usuarios finales.

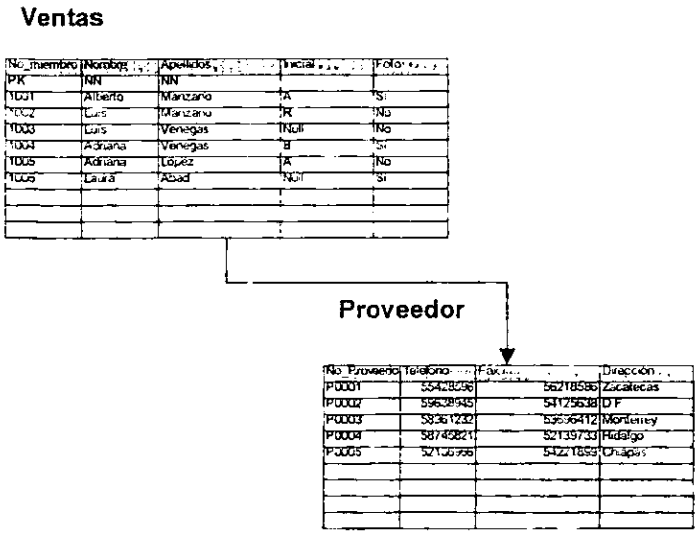


Fig. 3.1 Representación de Relaciones en el Modelo Relacional

Independencia Física. Con este modelo la forma en que se almacenan los datos no va a influir en la manipulación lógica, es decir que los usuarios que accedan a estos datos no tengan que modificar los programas por cambios en el almacenamiento físico. Esto es una gran ventaja, ya que se reduce el costo que produciría modificar los programas existentes en una organización.

Calidad de Diseño. Otro de los objetivos es poder juzgar la calidad del diseño. El modelo relacional proporciona los siguientes criterios para realizar buenas estructuras lógicas de datos:

- Cada evento debe almacenarse sólo una vez en la base de datos.
- La base de datos debe ser consistente con estas operaciones.
- La base de datos debe ser fácil de cambiar.

Como ya se menciono, el modelo relacional es uno de los modelos más utilizados actualmente y que ha tenido un mayor auge en el campo de las bases de datos, esto se da debido a ciertas razones dentro de las cuales están las siguientes:

- La interfaz tabular la pueden entender con facilidad los usuarios y los profesionistas de la computación y además sirve como herramienta de comunicación entre ellos.
- Es sencillo y fácil convertirlo en una implementación en un sistema de información.
- El modelo relacional proporciona un criterio formal para una buena representación de datos.

Los lenguajes que se pueden usar para accesar datos en una base de datos relacional se caracterizan por su interfaz sintáctica y su poder de selección. Uno de estos lenguajes, es el álgebra relacional, con la cual se pueden crear nuevas relaciones, con base en las existentes mediante operaciones de álgebra relacional.

3.2 ESTRUCTURA DEL MODELO RELACIONAL

El modelo relacional como ya se menciono anteriormente se representa a través de tablas, es decir, representa a la base de datos como una colección de relaciones, por esto se habla de tablas en el modelo relacional. Estas tablas están formadas por atributos y tuplas, a las tablas se les da el nombre de relaciones. Fig. 3.2. A continuación se da el significado de las partes que conforman parte del modelo relacional:

Atributos

Los atributos son cualidades adicionales o identificadores de una entidad. Los atributos son las columnas o campos en una tabla. Un ejemplo de atributos de una entidad. Tenemos una entidad llamada Empleado sus atributos serian el nombre, el apellido paterno y materno, edad, etc.

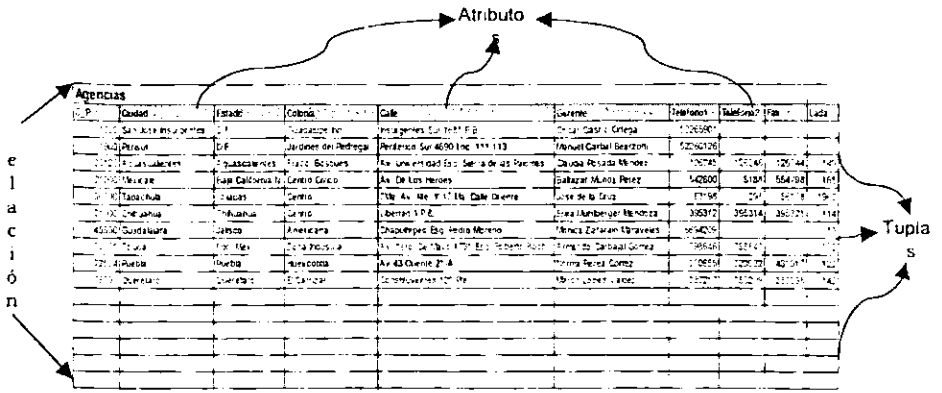


Fig. 3.2 Elementos del Modelo Relacional

Relaciones

Una entidad puede tener una relación con otra entidad. Las relaciones son definidas por llaves primarias y llaves secundarias, las cuales mantienen la integridad referencial. Las relaciones pueden señalarse en las columnas de las entidades, esto también depende del tipo de relación.

Una relación puede ser por intensión o por extensión, la intensión es la parte definitoria y estática de la relación, que se corresponde con la cabecera cuando la relación se percibe como una entidad.

La extensión es el conjunto de tuplas que, en un instante determinado, satisfacen el correspondiente esquema de relación, como el esquema de relación es invariante, su extensión varia en el transcurso del tiempo. En la fig. 3 3 se puede apreciar la intensión y la extensión

Tuplas

Estas tuplas son las filas y son las ocurrencias de la relación. Las tuplas de una relación no tienen un orden específico.

Cardinalidad

La cardinalidad es él numero de filas de una relación, mientras que el número de atributos de una entidad es el grado.

INTENSIÓN DE UNA RELACIÓN

Agencias (C_P = Código Postal, Ciudad = Ciudades, Estado = Estados,
Colonia = Colonias, Calle = Calles, Gerente = Gerentes,

EXTENSIÓN DE UNA RELACIÓN

AGENCIAS

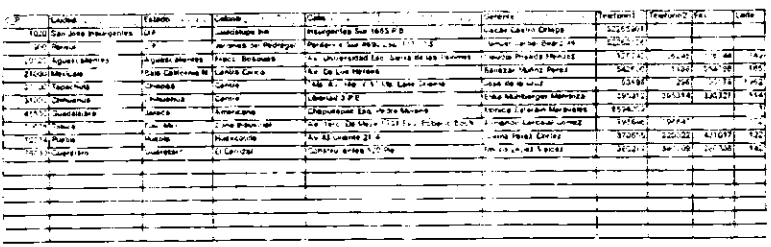
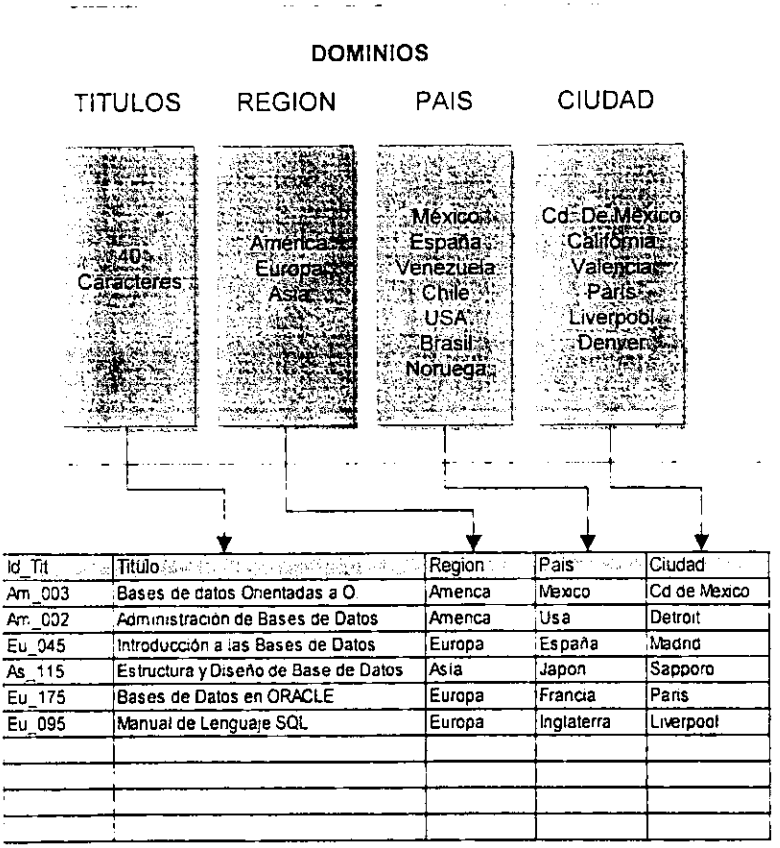


Fig. 3.3 Intensión y Extensión de una Relación

Dominio

El dominio es un conjunto finito de valores homogéneos y atómicos caracterizados por un nombre. Al hablar de valores homogéneos se refiere a que todos los elementos son del mismo tipo y atómicos porque son indivisibles en lo que al modelo se refiere, es decir, si se descompusiesen perderían la semántica a ellos asociados Fig. 3.4. Todos los dominios deben de tener un nombre por el cual se pueda hacer referencia a este y un tipo de datos



Id_Tit	Titulo	Region	Pais	Ciudad
Am_003	Bases de datos Orientadas a O	Americ a	Mexico	Cd de Mexico
Am_002	Administración de Bases de Datos	America	Usa	Detroit
Eu_045	Introducción a las Bases de Datos	Europa	España	Madrid
As_115	Estructura y Diseño de Base de Datos	Asia	Japon	Sapporo
Eu_175	Bases de Datos en ORACLE	Europa	Francia	Paris
Eu_095	Manual de Lenguaje SQL	Europa	Inglaterra	Liverpool

Fig. 3.4 Dominios de una Relación

Claves de Relaciones

La clave es una serie de atributos, cuyo valor identifica un renglón único en una relación. De esta forma dos renglones o n_tuplos no pueden tener el mismo valor de clave, lo que a veces se llama propiedad de unicidad. Hay diferentes formas de definir las claves que pueden poseer esta propiedad de unicidad

Claves de Relación (Claves Restringidas)

La clave restringida se define formalmente como un conjunto de atributos de relación encadenados para que las siguientes tres propiedades se mantengan en todo momento y para cualquier caso de la relación:

1. Unicidad, el conjunto de atributos que tiene un valor único para cada tupla en la relación.
2. No Redundancia, si un atributo se elimina de la serie, los demás no tendrán la propiedad de unicidad
3. Validez, No puede ser nulo ningún valor de atributo en la clave.

Una clave de relación se puede hacer con uno o muchos atributos. Las claves de relación también son lógicas y no tienen ninguna relación con la forma como sé accesan a los datos. Lo que especifica esta clave es que una relación tiene a lo sumo un renglón con un valor dado de la clave de relación.

Restricciones del Modelo Relacional

El modelo relacional, al igual que otros tipos de modelos que existen tienen ciertas restricciones. Es decir, estructuras u ocurrencias que no se permiten. Este tipo de restricciones se divide en dos, las restricciones inherentes y las restricciones de usuario. Estas restricciones se dan debido a que los datos se deben de adaptar a la estructura del modelo y se deben de cumplir con las restricciones de los usuarios.

Restricciones Inherentes

Dentro de las restricciones están las siguientes:

- No debe de haber dos tuplas iguales.
- El orden de las tuplas no es significativo.
- El orden de los atributos no es significativo.
- Cada atributo sólo puede tomar un único valor del dominio, de esta forma no se admiten los grupos repetitivos.

Otra restricción es la *Regla de Integridad de Entidad*, esta regla establece que ningún atributo que forme parte de la clave primaria de una relación puede tener un valor nulo, es decir, un valor desconocido o inexistente. Esta regla sólo se aplica a las claves primarias y no afecta a otro tipo de claves.

Restricciones de Usuario

Dentro de las restricciones de usuario esta la de *Integridad Referencial*, esta regla se refiere a que toda relación que tenga una clave ajena debe de concordar con la clave primaria de dicha relación.

La integridad referencial se especifica entre dos relaciones y sirve para mantener la consistencia entre tuplas de las dos relaciones. es decir, la restricción de integridad referencial establece que una tupla en una relación que haga referencia a otra relación deberá referirse a una tupla existente en esa relación.

3.3 LENGUAJES DEL MODELO RELACIONAL

El modelo relacional proporciona lenguajes para poder accesar relaciones Los lenguajes relacionales operan sobre conjuntos de tuplas, es decir, no son lenguajes de navegación sino de especificación.

Algebraicos, se caracteriza ya que los cambios de estado se especifican mediante operaciones donde los operadores son relacionales y el resultado que se obtiene es otra relación, este lenguaje recibe el nombre de Algebra Relacional.

3.3.1 ALGEBRA RELACIONAL

El álgebra relacional es un lenguaje procedural, es decir, es una colección de operaciones que sirven para manipular relaciones enteras. El álgebra relacional consiste en una serie de operadores, los cuales sirven para seleccionar tuplas de relaciones con el fin de especificar una consulta de la base de datos.

Los operadores del álgebra relacional se clasifican en dos grupos:

1. El primer grupo se refiere a las operaciones de la teoría matemática de conjuntos, entre estas operaciones están las siguientes: la Unión, la Intersección, la Diferencia y el Producto Cartesiano.

2. El segundo grupo son operaciones creadas específicamente para bases de datos relacionales y estos operadores son: la Selección, la Proyección y el Join.

Selección

La selección también puede ser nombrada como restricción, la operación selección sirve para seleccionar un subconjunto de las tuplas de una relación que satisfacen una condición de selección.

La operación selección se denota de la siguiente manera

$$\sigma_{\text{<Condición de selección>}}(\text{<Nombre de relación>})$$

El símbolo σ (sigma) denota el operador de selección y la condición de selección es una expresión booleana especificada en términos de los atributos de la selección. Los operadores que se pueden utilizar son: =, <, ≤, >, ≥, ≠. Las cláusulas se pueden conectar con los operadores booleanos *Y (AND), O (OR) y NO (NOT)* para formar una condición de selección general.

Por ejemplo, si se desea seleccionar las tuplas de todos los alumnos que estudian Informática y tienen un promedio de más de 8, la operación seleccionar queda de la siguiente forma:

$$\sigma_{(\text{cve\_carrera} = \text{Informática And promedio} \geq 8)}(\text{Alumnos})$$

El resultado de esta operación se puede ver en la Fig. 3.6

Alumnos

Id_Alumno	Carrera	Nombre	A_Pat	A_Mat	Promedio
91313953	Informática	LUCINA	HERRERA	GARCIA	8.56
91313954	Informática	MIGUEL	ECHEVERRIA	MORENO	10.00
91313955	Contaduría	ISORA RLANDA	VALENZUELA	ELENES	9.45
88542125	Administración	EVERARDO	LOPEZ	LOPEZ	9.75
88542126	I.M.E.	MARIO	LUJAN	MORA	8.00
88542127	Informática	JESUS MANUEL	BELTRAN	ARMENTA	7.55
91313959	I.M.E.	CLAUDIA	LANDAVAZO	GEORGE	8.13
91313960	Administración	JESUS	MENDOZA	PEDROZA	9.45
91312961	Informática	MARIA ANTONIETA	HERRERA	TAPIA	9.50
91312962	Informática	MOISES FERNANDO	CASTRO	JUAN	7.3
91312963	Administración	JOSÉ	GARCIA	BENITEZ	7.00
91313964	Informática	LEONARDO	CARMA	RODRIGUEZ	9.09
91312965	Contaduría	SERGIO LUIS	RADILLO	LLERA	8.0

Resultado de la Selección de Alumnos

Id_Alumno	Carrera	Nombre	A_Pat	A_Mat	Promedio
91313953	Informática	LUCINA	HERRERA	GARCIA	8.56
91313954	Informática	MIGUEL	ECHEVERRIA	MORENO	10.00
91313961	Informática	MARIA ANTONIETA	HERRERA	TAPIA	9.50
91313964	Informática	LEONARDO	CARMA	RODRIGUEZ	9.09

Fig 3.6 Operación Selección

Proyección

Con la proyección se construye una nueva relación a partir de alguna ya existente, es decir, sólo se seleccionan los atributos específicos de la relación existente y elimina los tuplos duplicados en la nueva relación formada.

La operación proyección se denota de la siguiente manera:

$$\pi_{\text{<Lista de atributos>}}(\text{<Nombre de la Relación>})$$

El simbolo π (pi) denota la operación de proyección y la <Lista de atributos> es una lista de atributos de la relación especificada por <nombre de la relación>. La relación que se crea contiene sólo los atributos especificados en la <Lista de atributos> y en el mismo orden que aparece en la lista.

Por lo general el número de tuplas que resulta de una relación de una operación de proyección es usualmente menor o igual al número de tuplas de la relación original. La operación de proyección no es conmutativa.

Por ejemplo, si se desea proyectar el nombre del alumno y la carrera de la relación alumnos, esta quedaría de la siguiente forma.

$$\pi \text{ nombre, carrera (Alumno)}$$

La proyección nos permite mostrar al usuario una vista de una relación con la información que a este solamente le interesa de dicha relación y de esta forma se facilita la comprensión de la información mostrada al usuario. El resultado de esta operación se puede ver en la Fig. 3.7

Alumnos

M_Alumno	Carrera	Nombre	A_Pat	A_Mat	Promedio
9[illegible]3	Informática	LUCINA	HERRERA	GARCIA	8.00
9[illegible]4	Informática	MIGUEL	ECHEVERRIA	MORENO	10.00
9[illegible]5	Contaduría	ISORA IRLANDA	VALENZUELA	FLORES	7.45
8854[illegible]25	Administración	EVERARDO	LOPEZ	LOPEZ	8.75
8854[illegible]26	TM E	MARIO	LUJAN	MORA	9.00
8854[illegible]27	Informática	JESUS MANUEL	BELTRAN	ARMENTA	7.50
9[illegible]9	TM E	CLAUDIA	LANDAVAZO	GEORGE	8.75
9[illegible]0	Administración	JESUS	MENDOZA	PEDROZA	9.45
9[illegible]1	Informática	MARIA ANTONIETA	HERRERA	TAPIA	6.90
9[illegible]2	Informática	MOISES FERNANDO	CASTRO	JUAN	7.25
9[illegible]3	Administración	JOSE	GARCIA	SENTIEZ	6.00
9[illegible]4	Informática	LEONARDO	GARCIA	RODRIGUEZ	9.00
9[illegible]5	Contaduría	SERGIO LUIS	RADILLO	LLERA	8.05

Resultado de la Proyección de Alumnos

Carrera	Nombre
Informática	LUCINA
Informática	MIGUEL
Contaduría	ISORA IRLANDA
Administración	EVERARDO
TM E	MARIO
Informática	JESUS MANUEL
TM E	CLAUDIA
Administración	JESUS
Informática	MARIA ANTONIETA
Informática	MOISES FERNANDO
Administración	JOSE
Informática	LEONARDO
Contaduría	SERGIO LUIS

Fig. 3 7 Operación Proyección

Reunión

La operación reunión permite combinar dos o más relaciones en una sola Se requiere que se seleccionen los atributos para igualar tuplos en las relaciones. Los tuplos en diferentes relaciones pero con el mismo valor de los atributos correspondientes se combinan en un tuplo único en el resultado de la relación.

Esta operación es muy importante en una base de datos relacional que contenga más de una relación, ya que permite procesar los vínculos entre las relaciones. Esta operación se denota de la siguiente forma.

$$R \times_{\text{<Condición de reunión>}} S$$

Donde el resultado de la operación de la reunión es una relación Q con n + m atributos, es decir, Q tiene una tupla por cada combinación de tuplas (una de R y una de S), siempre y cuando se cumpla la condición de la operación de reunión. En la reunión siempre aparecerá la combinación de tuplas que satisfagan la condición de dicha reunión.

Por ejemplo, si se desea obtener el nombre de las publicaciones y su titulo de las relaciones de Publicaciones y Títulos, la operación Reunión quedara de la siguiente forma:

$$\text{Publicaciones} \times_{\text{<pub\_id = Pub\_id>}} \text{Títulos}$$

El resultado de esta operación se puede ver en la Fig. 3.8

Publicaciones

Pub_Id	Pub_Name	Ciudad	Fecha
1389	Scootney Books	Boston	25/05/99
0736	GGB&R	Berkley	14/02/99
0877	Lucerne Publish	Paris	31/07/97
1248	Lucerne Publish	Paris	1/06/98
2589	Iberoamerica	Mexico	12/02/96
3598	New Moon Books	Washington	13/09/94
8694	Promua	México	25/03/97

Titulos

Ta_Id	Titulo	Pub_Id
BU_1032	Oracle/SQL Professional	1389
PC_1035	SQL the Structured Query Language	1389
BU_2075	Database Design Methodology	0736
PS_2091	Object Oriented Software	0736
PS_2106	Formal Techniques for Data Base Design	0736
BF_0568	Conceptos de Base de Datos	2589
DS_0480	Diseño de BD en SQL Server 6.5	8359

Resultado de la Reunión
de Publicaciones y Titulos

Pub_Id	Pub_Name	Ciudad	Fecha	Tit_Id	Titulo
1389	Scootney Books	Boston	25/05/99	BU_1032	Oracle/SQL Professional
1389	Scootney Books	Boston	25/05/99	PC_1035	SQL the Structured Query Language
0736	GGB&R	Berkley	14/02/99	BU_2075	Database Design Methodology
0736	GGB&R	Berkley	14/02/99	PS_2091	Object Oriented Software
0736	GGB&R	Berkley	14/02/99	PS_2106	Formal Techniques for Data Base Design
2589	Iberoamerica	Mexico	12/02/96	BF_0568	Conceptos de Base de Datos

Fig. 3.8 Operación Reunión

Unión

La unión de dos relaciones compatibles en un esquema es otra relación definida sobre el mismo esquema de relación, donde la extensión estará constituida por las tuplas que pertenezcan a la relación R o a la relación S

La operación unión se denota de la siguiente manera $R \cup S$, es una relación que incluye todas las tuplas que están en R o en S o en ambas. En esta relación las tuplas repetidas se eliminan. Por ejemplo, si se desea obtener la unión de las relaciones de Publicaciones y Titulos, se denotaría de la siguiente forma.

Publicaciones $\cup$ Titulos

El resultado de esta operación se puede ver en la Fig. 3.9.

Publicaciones

Pub_Id	Pub_Name	Ciudad	Fecha
1389	Scootney Books	Boston	25/05/99
0736	GGB&R	Berkley	14/02/99
0877	Luceme Publish	Paris	31/07/97
1248	Luceme Publish	Paris	1/06/98
2589	Iberoamerica	Mexico	12/02/96
3556	New Moon Books	Washington	15/09/94
8694	Promua	Mexico	25/03/97

Titulos

Tit_Id	Titulo	Pub_Id
BU_1032	Oracle/SQL Professional	1389
PC_1035	SQL the Structured Query Language	1389
BU_2075	Database Design Methodology	0736
PS_2091	Object Oriented Software	0736
PS_2106	Formal Techniques for Data Base Design	0736
BF_0568	Conceptos de Base de Datos	2589
DG_0480	Diseño de BD en SQL Server 6.5	6359

Resultado de la Unión de Publicaciones y Titulos

Pub_Id	Pub_Name	Ciudad	Fecha	Tit_Id	Titulo
1389	Scootney Books	Boston	25/05/99	BU_1032	Oracle SQL Professional
0736	GGB&R	Berkley	14/02/99	BU_2075	Database Design Methodology
0877	Lucerne Publish	Paris	31/07/97	PF_2158	Sistemas de Redes
1248	Lucerne Publish	Paris	1/06/98	PF_8974	Redes WAN
2589	Iberoamerica	Mexico	12/02/96	BF_0568	Conceptos de Base de Datos
3598	New Moon Books	Washington	13/09/94	EF_5148	Fundamentos de Unix (HP)
8634	Promua	Mexico	25/03/97	TO_1254	Admon De Redes en Unix
6359	Edic. Limitadas	Mexico	25/07/99	DG_0480	Diseño de BD en SQL Server 6.5

Fig. 3.9 Operación Unión

Intersección

La intersección de dos relaciones compatibles en su esquema es otra relación definida sobre el mismo esquema de relación, cuya extensión estará definida por las tuplas que pertenezcan a ambas relaciones. La operación intersección se denota de la siguiente forma R ∩ S, es una relación que incluye las tuplas que están tanto en R como en S. Por ejemplo, la intersección de las relaciones de Publicaciones y Titulos quedaria de la siguiente forma:

Publicaciones ∩ Titulos

El resultado de esta operación se puede ver en la Fig. 3.10

Publicaciones

Pub_Id	Pub_Name	Ciudad	Fecha
1389	Scootney Books	Boston	25/05/99
0736	GGB&R	Berkley	14/02/99
0877	Lucerne Publish	Paris	31/07/97
1248	Lucerne Publish	Paris	1/06/98
2589	Iberoamerica	Mexico	12/02/96
3598	New Moon Books	Washington	13/09/94
8634	Promua	Mexico	25/03/97

Titulos

Tit_Id	Titulo	Pub_Id
BU_1032	Oracle SQL Professional	1389
PC_1035	SQL the Structured Query Language	1389
BU_2075	Database Design Methodology	0736
PS_2091	Object Oriented Software	0736
PS_2106	Formal Techniques for Data Base Design	0736
BF_0568	Conceptos de Base de Datos	2589
DG_0480	Diseño de BD en SQL Server 6.5	6359

Resultado de la Intersección de Publicaciones y Titulos

Pub_Id	Pub_Name	Ciudad	Fecha	Tit_Id	Titulo
1389	Scootney Books	Boston	25/05/99	PC_1035	SQL the Structured Query Language
0736	GGB&R	Berkley	14/02/99	BU_2075	Database Design Methodology
2589	Iberoamerica	Mexico	12/02/96	BF_0568	Conceptos de Base de Datos

Fig. 3.10 Operación Intersección

Diferencia

La diferencia de dos relaciones compatibles en un esquema es otra relación definida sobre el mismo esquema de relación, cuya extensión estará constituida por el conjunto de tuplas que pertenezcan a una relación R, pero no a la otra relación S.

La operación de diferencia se denota de la siguiente manera R - S, es una relación que incluye todas las tuplas que están en R pero no en S. Por ejemplo, la operación de diferencia entre las relaciones de Publicaciones y Titulos o viceversa quedaría de la siguiente forma:

Publicaciones - Titulos ó Titulos - Publicaciones

El resultado de esta operación se puede ver en la Fig. 3.11

Publicaciones

Pub_Id	Pub_Name	Ciudad	Fecha
1389	Scootney Books	Boston	25/05/99
0736	GGB&R	Berkley	14/02/99
0877	Lucerne Publish	Paris	31/07/97
1248	Lucerne Publish	Paris	17/06/98
2589	Iberoamerica	Mexico	12/02/96
2558	New Moon Books	Washington	13/09/94
8684	Promua	Mexico	25/03/97

Titulos

Tit_Id	Titulo	Pub_Id
BU_1032	Oracle/SQL Professional	1389
PC_1035	SQL the Structured Query Language	1389
BU_2075	Database Design Methodology	0736
PS_2091	Object Oriented Software	0736
PS_2106	Formal Techniques for Data Base Design	0736
BF_0568	Conceptos de Base de Datos	2589
DG_0480	Diseño de BD en SQL Server 6.5	6359

Resultado de la Diferencia de Publicaciones y Títulos

Pub_Id	Pub_Name	Ciudad	Fecha	Tit_Id	Título
0877	Lucerne Publish	Paris	31/07/97		
1248	Lucerne Publish	Paris	1/06/98		
3598	New Moon Books	Washington	13/09/94		
8594	Proirrua	Mexico	25/03/97		

Fig. 3.11 Operación Diferencia

Producto Cartesiano

El producto es una relación cuyo esquema estará definido sobre la unión de los atributos de ambas relaciones y cuya extensión estará constituida por las tuplas formadas, concatenando cada tupla de la relación S con cada una de las tuplas de la relación R.

La operación del producto cartesiano se denota de la siguiente manera S x R. es una relación en la que se combinan las dos relaciones. El producto cartesiano casi nunca se usa como operación significativa por si sola, por lo general el producto cartesiano siempre esta seguido de la operación seleccionar.

Por ejemplo, si se desea utilizar la operación del producto cartesiano en la relaciones de Publicaciones y Títulos, esto quedaría de la siguiente forma:

Publicaciones x Títulos

El resultado de esta operación se puede ver en la Fig. 3.12

Publicaciones

Pub_Id	Pub_Name	Ciudad	Fecha
1389	Scootney Books	Boston	25/05/99
0736	GGB&R	Berkley	14/02/99
0877	Luceme Publish	Paris	31/07/97
1248	Luceme Publish	Paris	1/06/98
2589	Iberoamerica	Mexico	12/02/96
3698	New Moon Books	Washington	13/09/94
8634	Promua	Mexico	25/03/97

Titulos

Tit_Id	Titulo	Pub_Id
BU_1032	Oracle/SQL Professional	1389
PC_1035	SQL the Structured Query Language	1389
BU_2075	DataBase Design Methodology	0736
PS_2091	Object Oriented Software	0736
PS_2106	Formal Techniques for Data Base Design	0736
BF_2568	Conceptos de Base de Datos	2589
DG_0468	Diseño de BD en SQL Server 6.5	6359

Resultado del Producto Cartesiano de Publicaciones y Titulos

Pub_Id	Pub_Name	Ciudad	Fecha	Tit_Id	Titulo
1389	Scootney Books	Boston	25/05/99	BU_1032	Oracle/SQL Professional
1389	Scootney Books	Boston	25/05/99	PC_1035	SQL the Structured Query Language
1389	Scootney Books	Boston	25/05/99	BU_2075	Database Design Methodology
0736	GGB&R	Berkley	14/02/99	PS_2091	Object Oriented Software
0736	GGB&R	Berkley	14/02/99	PS_2106	Formal Techniques for Data Base Design

Fig. 3.12 Operación Producto Cartesiano

División

La división de dos relaciones es otra relación cuya extensión estará constituida por las tuplas que al completarse con las tuplas de la segunda relación permiten obtener la primera. En general la operación división se aplica a dos relaciones y se denota de la siguiente manera $R(Z) \div S(X)$, donde $X \subseteq Z$. Sea Y = Z - X; es decir, sea Y el conjunto de atributos de R que no son atributos de S.

La división se puede expresar en función de la proyección, del producto cartesiano y de la diferencia de la siguiente forma:

$$\pi_Y(R) - \pi_Y ((S \times T) - R)$$

3.4 ARQUITECTURA CLIENTE / SERVIDOR

La arquitectura cliente/servidor se creó para manejar los nuevos entornos de cómputo en los que un gran número de computadoras personales, estaciones de trabajo, servidores de archivos, impresoras y otros equipos están interconectados a través de una red.

La arquitectura cliente/servidor se está incorporando cada vez más en los paquetes de SGBD comerciales conforme se van orientando hacia la distribución, esto consiste en dividir el software en dos niveles cliente y servidor, de esta forma sé esta reduciendo la complejidad que esto ocasionaba.

Las organizaciones pueden empezar el desarrollo usando el modelo cliente/servidor en una de las dos formas básicas; migrando solicitudes legando preferiblemente en piezas manejables; o desarrollando nuevas solicitudes (incluyendo sistemas criticos de misión) que fueron previamente no factible con la tecnología.

Un método efectivo de continuar con la arquitectura cliente/servidor es involucrando a expertos externos para el primer proyecto cliente/servidor y una vez construido el proyecto, se debe de ir entrenando a miembros del equipo de la organización en asuntos de este tipo de arquitectura. Con esto se puede obtener experiencia de las compañías externas.

Algunas de las caracteristicas con las que debe contar tanto la parte del servidor como del cliente son:

Plataforma del Servidor

1 Base de Datos. Al seleccionar el SGBD se debe de tomar en cuenta el número de usuarios, la programabilidad de la base de datos del servidor, la integridad referencial, la seguridad interna y los permisos, capacidad administrativa, la recuperación y la recuperación en caso de desastre.

2. Plataforma. Al considerar la plataforma se debe de ver el costo, el apoyo por parte del proveedor, el costo del mantenimiento y el récord de los datos seleccionados del servidor. Dentro de las caracteristicas deseables dentro de una plataforma están las de falta de tolerancia, múltiples CPU´s y discos reflejantes.

3. Sistema Operativo del Servidor. Las caracteristicas comunes son: multiusuario, escalabilidad, multicompilador. Otra consideración es la plataforma de apoyo y la estabilidad del Sistema Operativo.

4. Tarjeta de Comunicación/Software. Esta debe de ser configurable para apoyar el máximo número de usuarios y protocolos de red.

Plataforma del Cliente

1. Interfaz de Programación (API). Este tipo de interfases generalmente se proporciona con la base de datos del servidor. Algunas otras consideraciones son:
   - Accesar a diferentes bases de datos del servidor,
   - Poder ejecutar códigos de programación,
   - Flexibilidad y portabilidad.

2. Herramientas de Interfaz para el Usuario. Las consideraciones incluyen lo siguiente:
   - Facilidad de uso y amigabilidad para el usuario,
   - Diccionario central de datos,
   - Costos de tiempo y
   - Texto basado contra GUI's.

3. Sistema Operativo del Cliente. Las consideraciones que debe de incluir son el tipo de interfaces de usuario y multiusuario contra los requerimientos del usuario solo.

4. Tarjetas de Comunicación/Software. Las consideraciones incluyen la elección de la red de arquitectura y la disponibilidad de los componentes de reemplazo. Cabe señalar que el tiempo de respuesta es afectado por esta selección.

En la Fig. 3.13 se puede ver la arquitectura Cliente/Servidor

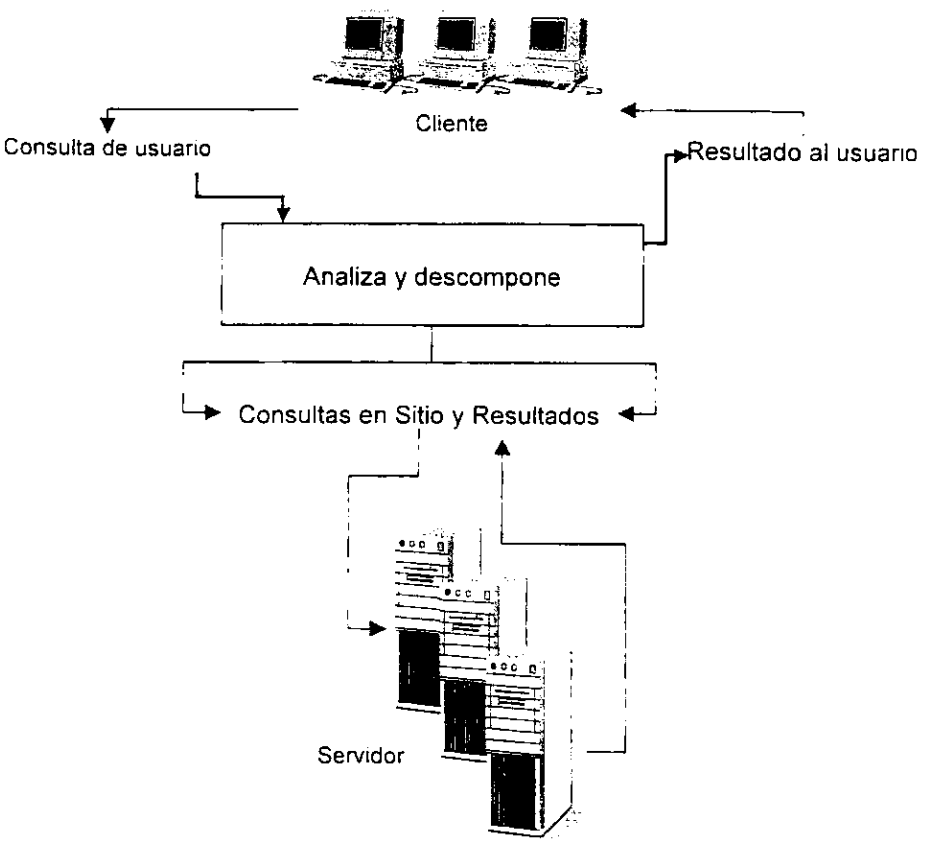


Fig. 3.13 Arquitectura Cliente/Servidor

3.5 LENGUAJE SQL

El álgebra relacional es imprescindible para entender los diferentes tipos de consultas que se pueden especificar en una base de datos relacional. Básicamente el álgebra relacional se clasifica dentro del rango de los lenguajes de consulta de alto nivel aunque existen muy pocos lenguajes en los SGBD que se basan en el álgebra relacional. El lenguaje más conocido comercialmente es el SQL el cual se analizara en esta parte.

El lenguaje SQL originalmente se llamaba SEQUEL (Structured English Query Language), el cual se diseñó e implemento por IBM Research como interfaz para un sistema experimental de base de datos relacional. Actualmente ANSI (American Standards Institute) e ISO (International Standards Organization) han realizado versiones estándar de SQL.

SQL emplea los términos de table (tabla), row (fila) y column (columna) en vez de relación, tupla y atributo respectivamente, esta es la estructura que maneja SQL. Básicamente existen tres tipos de sentencias en SQL: sentencias de definición, sentencias de manipulación y sentencias de control. Existen muchas sentencias para cada una de los diferentes tipos de estas, pero solo se explicarán aquellas sentencias que son más útiles tanto para la creación, la manipulación o control de una base de datos relacional.

Sentencias de Definición

Dentro de estas sentencias están las de creación de elemento como *Create table, Create Domain, Create View, etc.*; las de modificación para una estructura son: *Alter Table, Alter Domain, etc.* y por último las sentencias con las que se puede borrar son: *Drop Table, Drop Schema, etc.*

Create Table

La sentencia Create Table sirve para crear una tabla dentro de una base de datos, es decir sirve para especificar una nueva relación dándole un nombre y especificando sus

atributos y restricciones. Los atributos se especifican primero y a cada uno se le asigna un nombre, un tipo de datos para especificar su dominio de valores y si es necesario algún tipo de restricciones.

Con esta sentencia también se pueden crear las restricciones de clave, de integridad de entidades y de integridad referencial. Los tipos de datos disponibles para los atributos están los numéricos, los de cadena de caracteres, los de cadena de bits y los de fecha y hora.

Los tipos de datos de cadena de caracteres poseen una longitud fija (char (n) donde n es el número de caracteres) o variable (varchar (n), donde n es el número máximo de caracteres).

Los tipos de datos de bit tienen una longitud fija (bit (n)), o variable (bit varying (n), donde n es el número máximo de bits). El número por omisión de n, en una cadena de caracteres o de bits es de uno. En el siguiente ejemplo, se puede ver el código para crear una tabla (relación) en un SGBDR por medio del lenguaje SQL.

```
CREATE TABLE [database.[owner].]table_name
(       {col_name column_properties [constraint [constraint [...constraint]]]
        | [[,] constraint]}
                [[,] {next_col_name | next_constraint}...])
[ON segment_name]
```

donde:

table_name

Es el nombre de la nueva tabla que se va a crear. Este nombre debe ser unico y no se debe de repetir en la misma base de datos.

col_name

Es el nombre de la columna en una tabla. Los nombres de las columnas pueden estar conformados por reglas para poder identificarlas dentro de una tabla

Tipo de datos

En esta parte se especifica el tipo de datos que se de sea para una columna Estos tipos de datos los define el usuario.

```
if exists (select 1
        from  sysobjects
        where  name = 'asistentes'
        and   type = 'U')
  drop table asistentes
go
create table asistentes
(
        id_gral         char(20)        not null,
        nombre          char(30)        null,
        a_paterno       char(30)        null,
        a_materno       char(30)        null,
        fecha_reg       char(10)        null,
        empresa         char(50)        null,
        puesto          char(25)        null,
        tel_casa        char(12)        null,
        tel_oficina     char(12)        null,
        cve_plaza       int             null,
        fecha           char(10)        null,
        status          char(2)         null,
        contador        int             null
)
go
sp_primarykey asistentes, id_gral
go
CREATE  UNIQUE  INDEX asistentes_pk ON asistentes(id_gral)
GO
CREATE  INDEX asistentes_fk1 ON asistentes(fecha_reg)
```

```
GO
CREATE  INDEX asistentes_fk2 ON asistentes(hora)
GO
CREATE  INDEX asistentes_fk3 ON asistentes(cve_plaza)
GO
CREATE  INDEX asistentes_fk4 ON asistentes(fecha)
GO
sp_foreignkey evento, plaza, cve_plaza
go
sp_foreignkey asistentes, evento, cve_plaza, fecha
go
```

En el ejemplo anterior se puede ver la sentencia de drop table, la cual nos permite borrar una tabla de la base de datos y la sentencia create table, así como, la forma como se crean indices y las llaves primarias y foráneas de una tabla.

Alter Table

Con la sentencia Alter table se puede modificar una tabla de una base de datos. Las acciones que se pueden realizar con esta sentencia son las de adición o eliminación de una columna, la modificación de la definición de una columna y la adición o eliminación de restricciones de la tabla.

En el siguiente ejemplo, se puede ver el código de la sentencia Alter Table.

```
ALTER TABLE estudiantes add
        (sht number/4,1) null,
         swt number(4,1) null)
```

En el ejemplo anterior se agregaron dos nuevas columnas a la tabla de estudiantes. Con esta sentencia también se puede modificar uno, dos o todos los campos de la tabla, como se puede ver en el siguiente ejemplo.

ALTER TABLE nombre_tabla **MODIFY**

(columna-nombre tipo-dato [NULL | NOT NULL],
(columna-nombre tipo-dato [NULL | NOT NULL],

.

.

(columna-nombre tipo-dato [NULL | NOT NULL],

Sentencias de Manipulación

Las sentencias de manipulación permiten al usuario insertar, borrar o modificar los datos de una base de datos, también permiten estas sentencias la proyección de los datos.

Consulta en SQL

Una consulta en SQL puede constar de un máximo de seis cláusulas, pero sólo son obligatorias las dos primeras, Select y From. La forma como se deben de especificar las cláusulas es la siguiente:

Select <Lista de atributos>
From <Lista de tablas>
[Where <Condición>]
[Group by <atributo(s) de agrupación>]
[Having <Condición de agrupación>]
[Order By <Lista de atributos>]

SELECT. La cláusula Select indica que atributos o funciones se van a obtener de una consulta a la base de datos

FROM. La cláusula From especifica todas las relaciones que se necesitan en la consulta, incluidas las relaciones reunidas, pero no las que se requieren en consultas anidadas.

WHERE. La cláusula Where especifica las condiciones para seleccionar tuplas de las relaciones especificadas en la cláusula From.

GROUP BY. Con Group By se especifica los atributos de agrupación.

HAVING. Con la cláusula Having se especifica una condición que deben de cumplir los grupos seleccionados, no las tuplas individuales.

ORDER BY. Con la cláusula Order By se especifica un orden para presentar el resultado de una consulta, esta puede ser ascendente o descendente.

Algunos ejemplos de la orden Select son:

```
SELECT [Nombre], [Apellido_Pat]
FROM Empleados

SELECT Titulo, Count(Titulo)
FROM Empleados
WHERE Departamento = 'Producción'
GROUP BY Titulo HAVING Count(Titulo) > 50

SELECT [Nombre], [Apellido_Pat]
FROM Empleados
ORDER BY [Apellido_Pat] DESC
```

Como se menciona anteriormente existen tres órdenes en SQL para poder actualizar la base de datos: *Insert* (Insertar), *Delete* (Borrar) y *Update* (Modificar), a continuación se explicara cada una de estas.

INSERT

La orden Insert sirve para añadir una sola tupla a una relación. Para llevar a cabo esto se debe de especificar el nombre de la relación y una lista de los valores de la tupla. Los valores deberán listarse en el mismo orden en que se especificaron los atributos.

A continuación se presentan algunos ejemplos de la instrucción para insertar una tupla en una relación.

INSERT INTO Clientes (Id_Cli, Nombre_Cli, ApellidoPat_Cli, Direccion, Telefono, Fax, Fecha_Ing)
values (10000, "JOSE LUIS", "LOPEZ", "CALLE No. 8 CENTRO", "56358996", "5896521", "19970520")

INSERT INTO Clientes values (10000, "JOSE LUIS", "LOPEZ", "CALLE No. 8 CENTRO", "56358996", "5896521", "19970520")

INSERT INTO Clientes (id_cliente, nombre, apellidos, direccion, tel_casa, tel_oficina, fecha_ingreso) values (10000, "JOSE LUIS", "LOPEZ", "CALLE No. 8 CENTRO", "56358996", "5896521", "19970520")

INSERT INTO Empleados
SELECT Trainees
FROM Trainees
WHERE Fecha_Ingreso < Now() - 30

INSERT INTO Cliente
SELECT *
FROM Clientes_Nuevos
WHERE Id_Cliente = "Nuevo"

DELETE

La orden Delete elimina las tuplas de una relación, esta orden cuenta con una cláusula Where para seleccionar las tuplas que se van a eliminar. Las tuplas se eliminan explícitamente de una sola tabla a la vez. Si se omite la cláusula Where se esta especificando que se deben de eliminar todas las tuplas de la relación.

A continuación se presentan algunos ejemplos para el borrado de tuplas en una base de datos mediante la orden Delete.

```
DELETE FROM Empleados[11]

DELETE FROM Empleados
WHERE Titulo = 'Trainee'

DELETE FROM Empleados, Payroll,
INNER JOIN Payroll
ON Empleados.Empleado_ID = Payroll.Empleado_ID
WHERE Titulo = 'Trainee'
```

Nota: Se debe tener mucho cuidado al utilizar esta sentencia, ya que al ejecutarse esta se puede perder toda la información de la tabla o base de datos.

UPDATE

La orden Update sirve para modificar los valores de los atributos en una o más tuplas seleccionadas, esta orden también cuenta con una cláusula Where y que sirve para seleccionar las tuplas que se van a modificar.

A continuación se presentan algunos ejemplos para modificar una tupla de una relación, mediante la orden Update

UPDATE Empleados

SET No_Reporte = 5

WHERE No_Reporte = 2

UPDATE Productos

SET Precio_Unit = Precio_Unit * 1 1

WHERE ID_Producto = 8

AND Descontinuado = No

CAPITULO 4
CASO PRACTICO

4.1. ANTECEDENTES DE LA ORGANIZACION

El área de Servicio a Clientes maneja el concepto de Células de Trabajo, que consiste en grupos de 10 equipos los cuales están divididos en Asesores, Integradores y Solucionadores, Fig. 4.1. Esto se creo con el fin de aumentar la calidad y oportunidad en el servicio al cliente, capacitando y reforzando la calidad del personal, incrementando los niveles de productividad y calidad en los procesos, reduciendo costos de operación, minimizando riesgos, automatizando y haciendo eficientes los flujos de trabajo, disminuyendo el uso de altos volúmenes de papel y facilitando el uso del sistema.

Para poder hacer posible la implementación del concepto de "Células de Trabajo", en el área de Servicio a Clientes se desarrollo una Interfaz Gráfica.

El desarrollo de la Interfaz Gráfica está basado en un conjunto de ventanas gráficas, algunas ventajas de esta herramienta son:

- Sencillez.
- Manejo de información estándar de manera oportuna y confiable.
- Ahorro de tiempo de acceso a información al sistema de información.
- Automatización de varios de sus procesos de uso común.

La Interfaz Gráfica interactúa con un emulador de pantallas de 3270 a través del cual se ejecutan transacciones del Sistema de Información de Servicio a Clientes, disparadas mediante iconos, botones y menús gráficos. Una vez que el Sistema de Información regresa la información del cliente, la interfaz se encarga de ordenarla y presentarla en ventanas gráficas con campos en español y destacando la información que se considera de mayor uso e importancia para llamar la atención del representante de Servicio a Clientes.

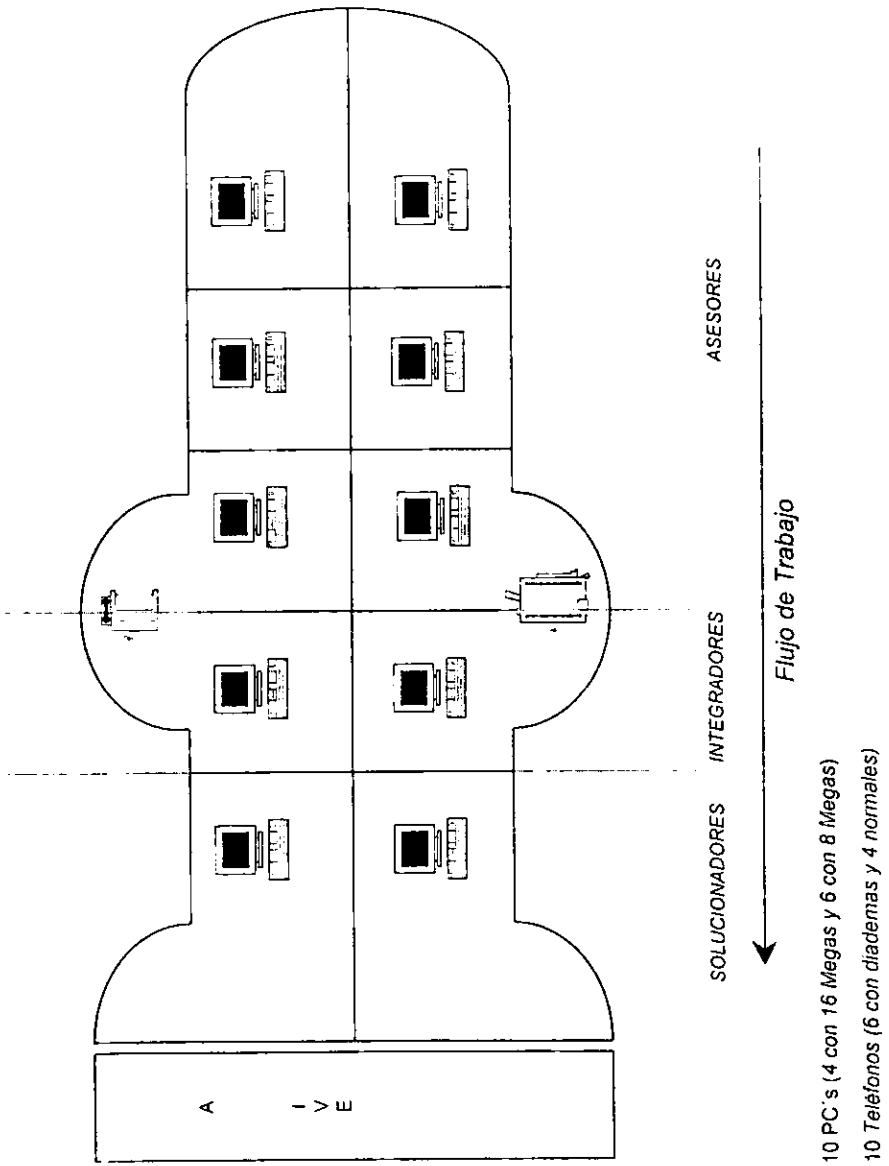


Fig. 4.1 Concepto Célula

La plataforma tecnológica que soporta a la Interfaz Gráfica está compuesta por estaciones de trabajo conectadas en una red local

Estaciones de Trabajo

- HP Vectra 486, 200 Mhz.
- 64 Mb RAM.
- 2.30 GB Disco Duro.
- Monitor 17" Ultra VGA.
- Tarjeta de Red Etherwist.
- Windows 95 en español.
- SNA Client V3.5 (Comunicación hacia el HOST)
- Extra for Windows 4.11 (Emulación 3270).
- Interfaz Gráfica de Servicio a Clientes.
- Excell 97
- Word 97

Red Local

Las estaciones de trabajo están conectadas a los switches, los cuales a su vez se conecta a los servidores. Estos servidores cuentan con una conexión al anillo de Token Ring, donde solo se transmite protocolo de SNA y en donde existe un router, el cual esta conectado a una antena de microondas que trasmite la señal hacia el Edificio Principal en donde por medio de routers son guiadas las peticiones de información al Host En la Fig. 4.2 se puede ver el Esquema de la Red de la Interfaz Gráfica.

Esquema de la Red de la Interfaz Gráfica

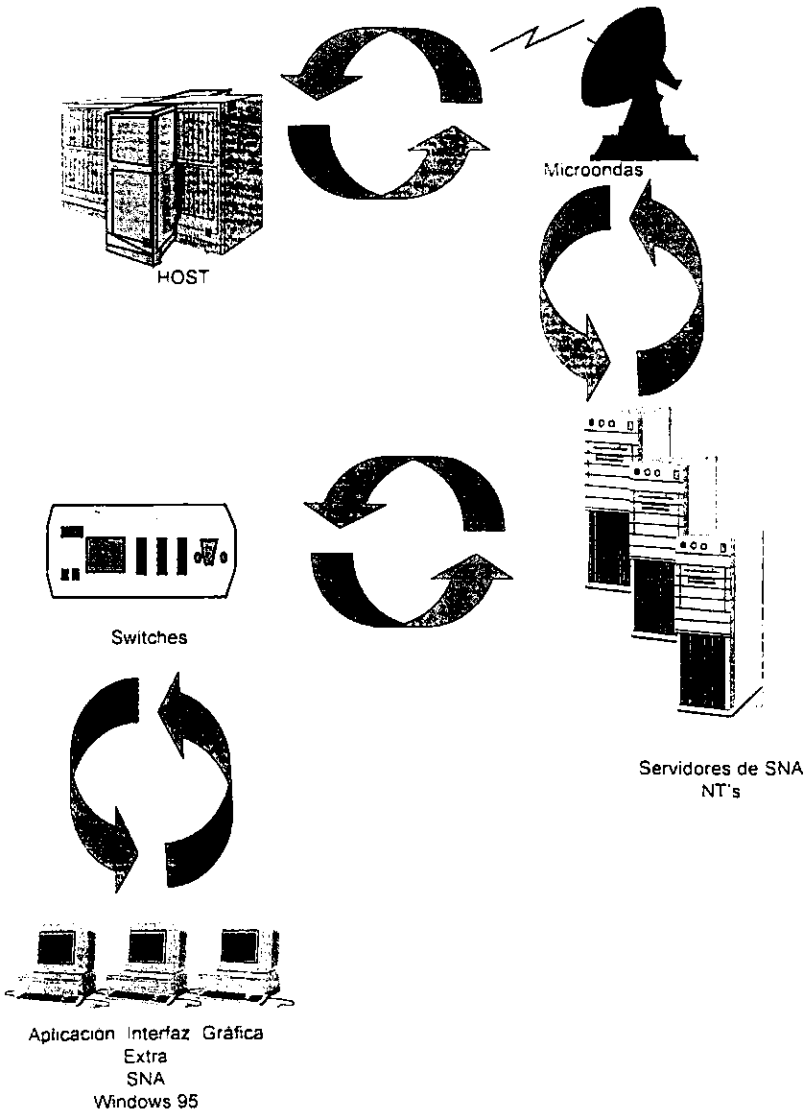


Fig. 4 2 Esquema de la Red

4.2 PLANTEAMIENTO DEL PROBLEMA

Debido a las necesidades del negocio de poder proporcionar un mejor servicio al cliente y poder unificar toda la infraestructura en un solo edificio, Servicio a Clientes requiere una plataforma que permita realizar todas las funciones del área de Planes de Financiamiento, tales como:

- Información General sobre los Planes Otorgados
- Alta de Autorizaciones
- Generación de Cotizaciones.
- Generación de Reportes
- Clasificación de Llamadas
- Historia de Contactos
- Información de Distribuidoras.
- Llamadas de salida

Esta plataforma debe trabajar bajo el concepto de célula. Actualmente el área de planes de financiamiento cuenta con los siguientes sistemas:

- Sistema de Planes de Financiamiento.
- Marcaje Personal.
- Control de Solicitudes
- El Emulador de Cotizaciones.

Sistema de Planes de Financiamiento

El Sistema de Planes de Financiamiento fue creado y diseñado bajo el concepto de célula, este sistema proporciona al cliente el servicio de post-venta, dentro sus principales funciones están:

1. Información General del Cliente, como Nombre, domicilio, teléfono, número de préstamo, cuenta de cargo, crédito inicial, etc.
2. Aclaraciones referente a sus prestamos.
3. Impresión de estado de cuenta, cartas facturas, etc.
4 Información general sobre los planes de financiamiento.

Actualmente esta siendo ocupado por 30 usuarios que tienen una capacidad de atender 35,000 llamadas.

Sistema de Marcaje Personal

El programa de marcaje personal surge a iniciativa de la Unidad de Negocios y como una necesidad de identificar que cliente llama, su potencialidad y cierre de operaciones.

Los objetivos que persigue este sistema son:

- Incremento adicional en la colocación de créditos.
- Evitar la perdida de clientes potenciales registrando de manera sistemática a los clientes que se comunican.
- Priorizar al cliente en función a sus potencialidad para colocar el crédito.
- Seguimiento a los clientes hasta el cierre de la venta.

Control de Solicitudes

Este sistema lleva un control sobre las solicitudes que se generan para obtener un préstamo. Dentro de las funciones del sistema están:

- Alta de solicitudes
- Consulta de datos (Estatus de Solicitudes)
- Generación de Reportes.

Emulador de Cotizaciones

El emulador de cotizaciones permite generar una cotización de los diversos planes con los que cuenta la institución. De acuerdo al tipo de plan, tipo de producto, modelo del producto, plazo, tipo de seguro y tasas.

Tanto el sistema de Marcaje Personal, Control de Solicitudes y Emulador de Cotizaciones se localizan en un centro diferente al Call Center, dentro de las peticiones del usuario es poder trasladar a todo el personal que atiende estos sistemas al Call Center Principal y poder aprovechar la infraestructura y tecnología con la que se cuenta actualmente.

4.3 SOLUCION DEL PROBLEMA

Para solucionar el problema de poder contar con un solo sistema que reúna todas las características necesarias para poder satisfacer las necesidades tanto del negocio como del cliente y poder aprovechar al máximo los recursos, se propone integrar la funcionalidad de los sistemas de Marcaje Personal, Control de Solicitudes y Emulador de Cotizaciones al sistema de Planes de Financiamiento.

Además de que el sistema de planes de financiamiento fue desarrollado bajo el concepto de célula, lo cual permitiría explotar toda la funcionalidad de este concepto. Otras de los beneficios serían:

- Utilizar una misma plataforma para el área de Planes de Financiamiento
- Aprovechar los recursos de Planes de Financiamiento.
- Centralizar todo en un solo centro de atención especializada en planes de financiamiento.
- Disminuir el número de llamadas
- Generación de reportes confiables.

- Generación de llamas de salida
- Contar con una sola base de datos robusta y con información confiable.

Nueva Funcionalidad del Sistema de Planes

La interfaz gráfica sería la misma que utiliza la gente de célula de planes de financiamiento, solamente que se le estarían agregando nuevas funcionalidades y el desarrollo de la Base de Datos.

Este sistema se dividirá en dos partes la Especialista (Pre-venta) y Generalista (Post-venta) validando desde su entrada si se es usuario de Especialista o Generalista. esto a través de la base de datos. Si el usuario es Especialista tendrá la opción de entrar como asesor o como llamadas de salida. Si el usuario es Generalista podrá entrar como asesor y solucionador. El usuario también podrá tener los dos privilegios.

El nuevo sistema podrá contabilizar todas las llamadas, ya sea por ejecutivos o por clientes potenciales. También se ofrecerá la consulta de:

- Datos demográficos de las distribuidoras.
- Planes de financiamiento de acuerdo a los productos.
- Convenios de las distribuidoras
- Montos de los seguros dependiendo de los modelos
- Cotizaciones de cualquier plan.

También se ofrece la generación de autorizaciones, la consulta de los datos demográficos de una cta. de cheques y la consulta de un préstamo vigente.

En la generación de una autorización se permitirá consultar, generar, registrar en la base de datos, dar de baja las autorizaciones para todos los planes. Otra de las opciones es la de realizar llamadas de salida.

Diseño de la Base de Datos

De acuerdo con las necesidades del negocio y principalmente a la funcionalidad que requiere el sistema se va a utilizar SQL Server 6.5 como el motor de la base de datos, a continuación se muestra cada una de las relaciones con la que va a contar la base de datos de planes de financiamiento.

Relación Autorizaciones

Esta relación guardara los datos referentes a las autorizaciones que se generen al día y sus atributos serán los siguientes:

Autorizaciones	
Distribuidora	char (80)
Marca	char (30)
Modelo	int
Cve_Producto	char (3)
Plazo	char (3)
Tipo_Cta	char (3)
Importe	char (20)
Num_Cta	char (20)
Num_Prestamo	char (20)
Num_Autorizaciones	char (20)
Tipo_Plan	char (80)
Fecha	char (8)
Hora	char (8)
Nombre_Cte	char (80)
Usuario	char (8)

Relación Confirma_Llamada

Esta relación permitirá llevar un control de las llamadas de salida que se realicen a los clientes y contara con los siguientes atributos:

Confirma_Llamada	
Fecha_Co	char (8)
Hora_Co	char (8)
Folio	char (20)
Num_Llamada	int
User_Id_Co	char (8)

Relación Detalle

Esta relación será un catalogo que contendrá las características principales de los productos tales como a que plan pertenecen, que tipo de producto es, su tasa, enganche mínimo, etc., en la siguiente estructura se puede ver todos los atributos con los que contara esta relación.

Detalle	
Cve_Esquema	char (2)
Cve_Plan	char (3)
Cve_Producto	char (2)
Cve_Seguros	int
Persona	char (2)
Tipo_NS	char (2)
Tipo_Tasa	char (2)
Plazo	int
Eng_Min	int
Empresa	char (5)
Tasa	float
Piso	float
Factor	float
Techo	float
Refinan	float
Eng_Max	float

Relación Distribuidora

Esta relación también será un catalogo que contendrá información acerca de las distribuidoras y sus atributos serán:

Distribuidora	
Id_Gral2	int
Cve_Marca	int
Num_Afiliacion	char (8)

Relación Estatus

La relación estatus será un catalogo que contendrá el estatus de las autorizaciones y los atributos de esta relación serán:

Estatus	
Cve_Estatus	int
Descripcion	char (30)

Relación Generales

Esta relación funcionara como catalogo ya que contendrá información de los clientes, ejecutivos, distribuidoras y sucursales. Los atributos con los que cuenta esta relación son los siguientes:

Generales	
Id_Gral	int
Nombre	char (100)
Apellido_Paterno	char (30)
Apellido_Materno	char (30)
Calle_Num	char (50)
Delegacion	char (50)
Colonia	char (50)
Cve_Estado	char (3)
Cve_Pais	char (3)
Cve_Gral	int

Relación Importe_Seguros

Esta relación será un catalogo de los importes de los seguros a través del cual al usuario se le facilitara conseguir el importe del seguro de algún producto en particular y los atributos de esta relación serán los siguientes:

Importe_Seguros	
Cve_BienProducto	char (7)
Modelo	int
Cve_Plazo	int
Importe	float

Relación Marcas

La relación también será un catalogo de que contendrá las diversas marcas de los productos que participen dentro de los planes de financiamiento, los atributos de esta relación serán:

Marcas	
Cve_Marcas	int
Descripcion	char (30)

Relación Plazo

Esta relación será un catalogo de los plazos a los que un plan puede someterse de acuerdo a las politicas de planes de financiamiento. La relación contara con los siguientes atributos:

Plazo	
Cve_plazo	int
Descripcion	char (20)

Relación Producto

Esta relación será un catalogo de los productos, la cual tiene la finalidad de proporcionar al usuario una mayor facilidad para localizar los diversos productos que participen en los diversos planes de financiamiento. Sus atributos serán los siguientes.

Producto	
Cve_BienProducto	char (7)
Modelo	int
Descripcion	char (80)
Precio_Min	float
Precio_Max	float
Cve_Producto	char (2)
Cve_Marca	int

Relación Publicidad

La relación será un catalogo de los diversos medios de publicidad que hay, esta relación servirá para que cuando se registre la llamada del cliente se guarde en la relación de llamadas el tipo de publicidad por la que el cliente se entero de los planes con los que cuenta la institución. Los atributos serán los siguientes:

Publicidad	
Cve_Publicidad	int
Descripcion	char (80)

Relación Registro_Contactos

Esta relación contendrá la información necesaria para poder consultar los contactos que se tienen con los clientes y poder proporcionar un mejor servicio a estos. La relación contara con los siguientes atributos:

Registro_Contactos	
Id_Gral	int
Cve_Estatus	int
Fecha_Insercion	char (8)
Hora_Insercion	char (8)
User_Id	char (8)
Fecha_Sig_Contacto	char (8)
Hora_Sig_Contacto	char (8)
Comentarios	char (150)

Relación Registra_Llamada

Esta relación contendrá los datos necesarios de los clientes y ejecutivos que llamen a Servicio a Clientes, es decir será una historia de contactos de los clientes y ejecutivos, los atributos serán:

Registra_Llamada	
Fecha	char (8)
Hora	char (8)
User_Id	char (8)
Cve_Llamada	int
Cve_Publicidad	int
Cve_Producto	char (2)
Cve_Plan	char (3)
Id_Cte	int
Id_Eje	int
Id_Dis	int
Id_Suc	int
Cve_Nissan	int
Origen	char (2)
Estado	char (2)
Cve_Referencia	char (3)
Cta_Referencia	char (20)
Comentarios	char (200)

Relación Seguros

Esta relación será un catalogo de los seguros y facilitara el alta de los créditos que se otorguen a los clientes y sus atributos serán:

Seguros	
Cve_seguros	int
Tipo_NS	char (2)
Cve_producto	char (2)
Cve_Enviar	char (1)

Relación Tasa_Lider

La relación de la tasa líder será un catalogo que contenga la tasa actual del dia y que servirá para realizar las cotizaciones de los productos. Los atributos con los que contara esta relación serán:

Tasa_Lider	
Fecha	char (8)
Tasa	float

Relación Teléfonos

Esta relación permitirá registrar los diversos teléfonos con los que un cliente o ejecutivo cuente y de esta forma se pueda contactar al cliente de forma más rápida. Sus atributos son los siguientes:

Telefonos	
Id_Gral	int
Cve_Tipo	int
Numero_Telefonico	char (10)

Relación Tipo_Esquema

Esta relación será un catalogo de los diversos esquemas en los que se pueda manejar un tipo de plan y sus atributos con los que contara serán:

Tipo_Esquema	
Cve_Esquema	char (2)
Descripcion	char (50)

Relación Tipo_General

La relación será un catalogo de los tipos de clientes que hablan a Servicio a Clientes, es decir una clasificación que puede ser cliente directo, ejecutivo, distribuidor, sucursal, etc. Y sus atributos serán

Tipo_General	
Cve_Gral	int
Descripcion	Char (50)

Relación Tipo_Planes

Esta relación será un catalogo de los diversos planes con los que cuente el área de Planes de Financiamiento y sus atributos serán:

Tipo_Planes	
Cve_Plan	char (3)
Descripcion	char (50)
Cve_Producto	char (2)

Relación Tipo_Producto

Esta relación también es un catalogo en los que se pueden clasificar por categorías los productos y contara con los siguientes atributos:

Tipo_Producto	
Cve_Producto	char (2)
Descripcion	char (50)

Relación Tipo_Seguros

La relación de tipo de seguros será un catalogo la cual indicara al usuario si el seguro es anual, de contado, multianual, etc. Y la relación contara con los siguientes atributos:

Tipo_Seguros	
Cve_Seguros	int
Descripcion	char (50)

Relación Tipo_Telefonos

Esta relación será un catalogo de los tipos de teléfonos que el usuario podrá identificar para registrar o actualizar los teléfonos de los clientes. Los atributos serán los siguientes:

Tipo_Telefonos	
Cve_Tipo	int
Descripcion	char (30)

Relación Tipo_Llamada

Esta relación será una clasificación de las llamadas por las que un cliente hable a Servicio a Clientes, los atributos de esta relación serán los siguientes:

Tipo_Llamada	
Cve_Llamada	int
Descripcion	char (80)
Acceso	char (3)

Relación Usuarios

Esta relación contendrá la información de todos los usuarios, la cual permitirá el acceso y los privilegios con los que podrá contar el usuario para utilizar el Sistema de Planes de Financiamiento. Los atributos con los que contara la relación serán:

Usuarios	
User_Id	char (8)
Nombre	char (75)
Aplicación	int
Equipo	int

En la Fig. 4.3 se puede ver el modelo de la base de datos con todas sus relaciones

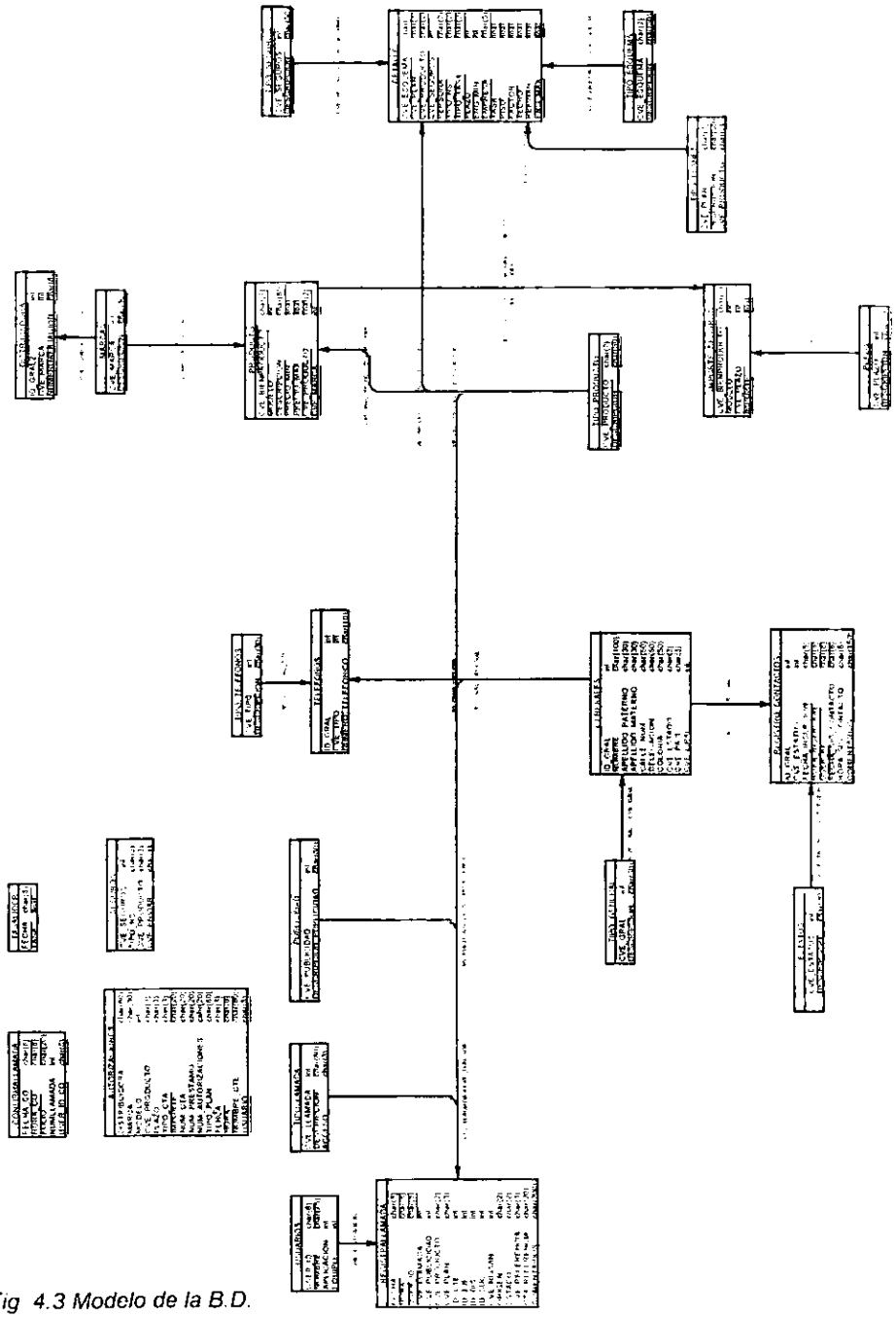


Fig 4.3 Modelo de la B.D.

Infraestructura

La infraestructura que deberá soportar al Sistema de Planes de Financiamiento estará compuesta por estaciones de trabajo conectadas en red local, servidor de base de datos y servidores de comunicación.

Servidor de Base de Datos

El servidor será exclusivo para este sistema se utilizara un servidor Compaq Prolaint con las siguientes características:

- Modelo Compaq Proliant 2500 P6/200 Mhz
- Dos Discos Duros, uno de 1 GB y el otro de 2 GB.
- Memoria RAM 64 MB
- Windows NT 4.00
- SQL Server 6.5

Servidores de Comunicaciones

Estos servidores son los que van a permitir comunicarnos con la matriz en donde se encuentra el Host, sus características son las siguientes:

- Modelo Compaq Proliant 4500 P5/100
- Disco duro de 4 GB
- Memoria RAM de 100 MB
- Windows NT 4.00
- SNA 3.0

Estaciones de Trabajo

Las estaciones de trabajo estarán configuradas de la siguiente manera:

- Equipos HP Vectra VE Pentium
- 64 MB RAM
- 2 GB Disco Duro

- Monitor 17" Ultra VGA
- Tarjeta de Red Etherwist
- Windows 95
- SNA Client V 3.5 (Comunicación hacia el Host)
- Extra for Windows 4.11 (Emulador 3270)
- Excel 97
- Word 97
- Sistema de Planes de Financiamiento

Todos estos serán los requerimientos que se necesitan para poder desarrollar el sistema y funcione de forma correcta. Cabe mencionar que el lenguaje que se utilizara para desarrollar el sistema es a través de Visual Basic Ver. 5.0. Los costos de infraestructura son muy bajos ya que se van a reutilizar tanto los servidores de comunicaciones como las PC's.

Implementación del Sistema

Como ya se había mencionado anteriormente el sistema se desarrollo en VB para poder aprovechar todos los recursos de este lenguaje de programación y el emulador Extra for Windows, con los cuales se desarrollaron conexiones o ligas que nos permiten extraer la información existente en las pantallas de Host en forma automática.

Esta información es distribuida en los campos correspondientes del Sistema de Planes de Financiamiento, con esto se estará evitando que los usuarios deban de aprender diferentes claves o códigos, la conexión del sistema con la base de datos se hace a través de ODBC. A continuación se muestran las funciones principales en las que el sistema hace uso de la Base de Datos.

Autorizaciones

Con el desarrollo de esta parte del sistema y el uso de la base de datos al usuario se le facilita el poder dar de alta de una forma sencilla y rápida una nueva autorización. En la Fig. 4 4 se puede ver como esta funcionando el sistema.

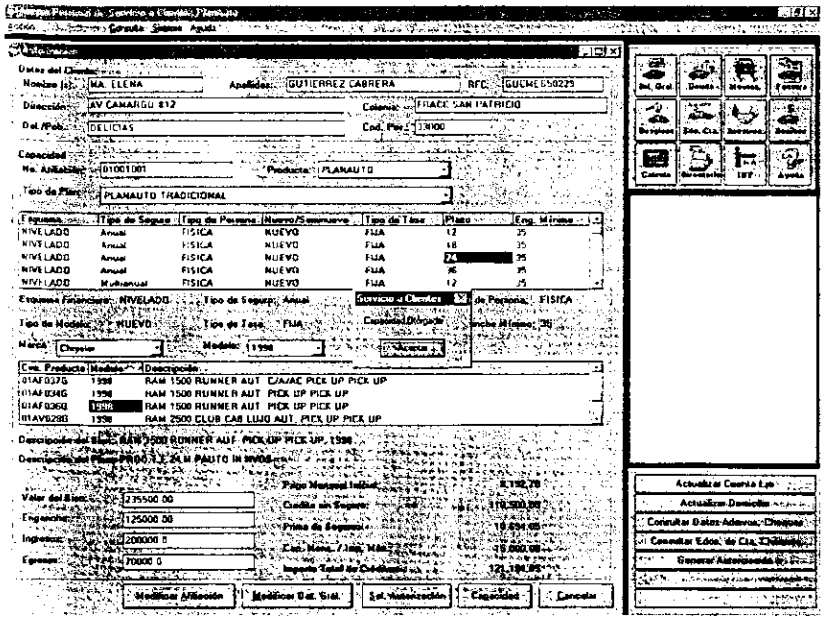


Fig. 4.4 Pantalla de Autorizaciones

Como se puede ver para dar de alta una autorización se hace uso principalmente de la información contenida en la base de datos la cual se presenta a través de combos y lo cual facilita al usuario, ya que este solo tiene que seleccionar los datos correctos para poder dar de alta la autorización.

Cotizaciones

La pantalla de cotizaciones permite al asesor la emisión inmediata de cotizaciones ya que el sistema permite realizar dicha tarea de una manera sencilla y eficiente, con lo cual se disminuye la canalización de esta tarea a una persona en particular. En la Fig. 4.5 se puede ver la pantalla de cotizaciones.

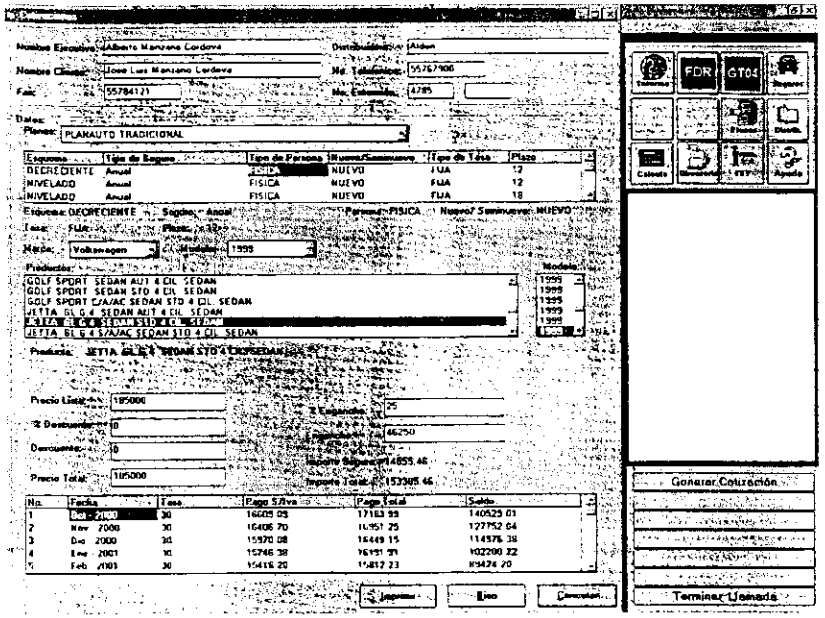


Fig. 4.5 Pantalla de Cotizaciones

Otras de las funcionalidades es la de poder mostrar información de tasas, seguros y planes mostrándola de forma clara y sencilla. En la Fig. 4.6 se muestran algunas de estas funciones que proporciona el Sistema de Planes de Financiamiento a través de la base de datos.

También cuenta con una historia de contactos lo cual facilita el desempeño de las actividades y permite tener de forma rápida información como el estatus de sus aclaraciones. El poder tener una sola base de datos con un sistema integrado con tres aplicaciones permite ofrecer un servicio rápido y eficiente al cliente.

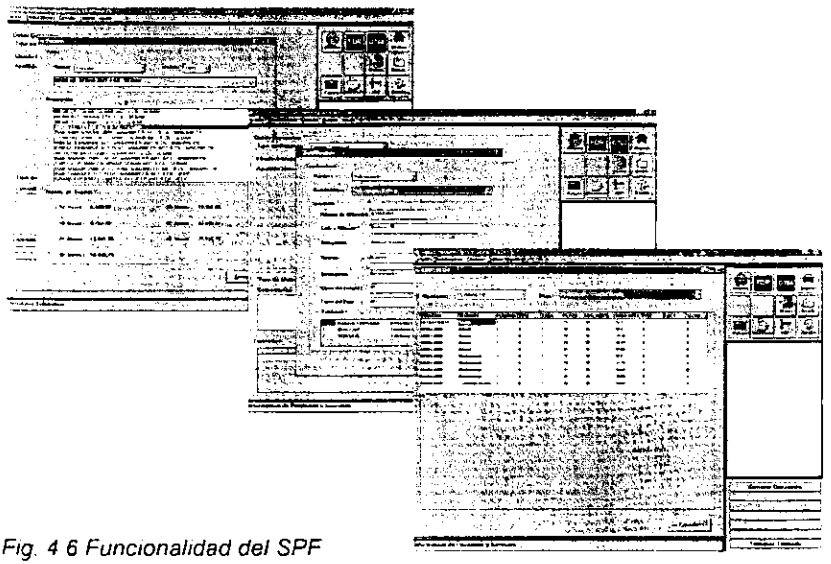


Fig. 4 6 Funcionalidad del SPF

Con el uso del sistema de planes de financiamiento y el de la base de datos se ha proporcionado un mejor servicio a los clientes, ya que facilita el trabajo del asesor al poder contar con información clara y precisa, así como poder realizar de forma automática y sin necesidad de utilizar claves tanto para generar cotizaciones como la alta de autorizaciones y lo principal es que cualquier asesor podrá realizar estas tareas

CONCLUSIONES

Como se puede ver el avance que han tenido los sistemas de base de datos ha sido muy rápido y vertiginoso, ya que no necesariamente al tener un SGDB poderoso se va a contar con información precisa y oportuna, ya que el buen desempeño de una base de datos se puede basar en varios puntos.

Uno de estos puntos son las necesidades del negocio, lo que requiere el usuario como información, el motor de la base de datos de acuerdo a la carga de información pero principalmente lo que hace que una base de datos sea eficiente y pueda proporciona al usuario información real y satisfactoria es la forma como se va a diseñar la base de datos desde el modelo que se va ha escoger ya sea relacional, orientado a objeto o entidad-relación hasta la creación de las tablas, campos y llaves, es lo que va a permitir contar con una buena base de datos que proporcione al usuario lo que el realmente requiera.

En cuanto al trabajo aquí expuesto se puede concluir lo siguiente: Al contar con un sistema de información integral, el asesor personal podrá dar una atención más completa al cliente, ya que le brindará el estatus de su solicitud, y en el caso de ser aceptada, autorizar el crédito. Con lo cual se evitará la transferencia de llamadas de manera innecesaria y se eliminará el uso de varios sistemas a la vez, ya que el Sistema de Planes de Financiamiento permitirá tener un registro de todas las llamadas de los distribuidores así como de clientes potenciales.

El asesor también estará facultado para hacer cambios en las características del producto, de acuerdo a su capacidad de pago. Por otro lado, el uso de la base de datos permitirá atender de manera personalizada a los Clientes Potenciales, ya que contará con un registro específico del estatus en el que se encuentra la venta

Al poder tener una base de datos que registra la información principal de los clientes se podrán realizar llamadas de salida y de esta forma proporcionar de manera clara la información que necesita cada cliente dependiendo del momento que viva cada

venta. Además, de que con esto se podrá llevar un mayor control de las ventas realizadas a través de los agentes.

De igual forma el asesor estará facultado para la emisión inmediata de cotizaciones, ya que el sistema le permitirá realizar dicha tarea de manera sencilla y eficiente, con lo cual se disminuirá la canalización de esta tarea a una persona adicional, mejorando en gran medida la atención a estas solicitudes del cliente.

Además de que el sistema proporcionara al asesor de manera sencilla y práctica la información correspondiente a seguros, planes y tasas. Proporcionando de esta forma una manera más fácil de trabajar al asesor y de esta forma mejorar el servicio a los clientes de Planes de Financiamiento.

BIBLIOGRAFIA

1 RAMEZ, Elmasri (1999). Sistemas de base de datos. conceptos fundamentales. México: Ed. Addison Wesley Iberoamericana.

2. PIATTINI, Mario G. (1998). Fundamentos y modelos de bases de datos. México: Ed. Alfaomega.

3. ATRE, Shakuntala. (1988). Técnicas de bases de datos. estructuración en diseño y administración. México: Ed. Trillas.

4. GONZALEZ, Alfons. (1999). SQL Server. programación y administración México: Ed. Alfaomega.

5. CAMPDERRICH, Benet. (1984). Técnicas de Bases de datos. Barcelona: Ed. Editores técnicos y asociados S.A.

6. DATE, C.J. (1996). Introducción a los sistemas de bases de datos. México: Ed. Addison Wesley Iberoamericana.

7. KRUGLINSKI, David. (1985). Sistemas de administración de bases de datos. México: Ed. MacGraw-Hill

8. DELOBEL, Claude. (1987). Bases de datos y sistemas relacionales. Barcelona Ed. Ediciones Omega.

9. DEEN. S M. (1987) Fundamentos de los sistemas de bases de datos Barcelona Ed. Gustavo Gill S A.

ANEXO 1

Estos son los scripts para generar las relaciones de la base de datos del Sistema de Planes de Financiamiento.

```
/* ==========================================================  */
/*  Database name:  Incremento                                 */
/*  DBMS name:      SQL Server                                 */
/*  Created on:     05/15/00                                   */
/* ==========================================================  */

if exists (select 1
          from  sysobjects
          where  name = 'generales'
          and   type = 'U')
  drop table generales
go

if exists (select 1
          from  sysobjects
          where  name = 'tipo_general'
          and   type = 'U')
  drop table tipo_general
go

if exists (select 1
          from  sysobjects
          where  name = 'telefonos'
          and   type = 'U')
  drop table telefonos
go

if exists (select 1
          from  sysobjects
          where  name = 'tipo_telefonos'
          and   type = 'U')
  drop table tipo_telefonos
go

if exists (select 1
          from  sysobjects
          where  name = 'publicidad'
          and   type = 'U')
  drop table publicidad
go

if exists (select 1
          from  sysobjects
```

```
        where  name = 'tipollamada'
         and   type = 'U')
  drop table tipollamada
go

if exists (select 1
         from  sysobjects
        where  name = 'usuarios'
         and   type = 'U')
  drop table usuarios
go

if exists (select 1
         from  sysobjects
        where  name = 'registro_contactos'
         and   type = 'U')
  drop table registro_contactos
go

if exists (select 1
         from  sysobjects
        where  name = 'estatus'
         and   type = 'U')
  drop table estatus
go

if exists (select 1
         from  sysobjects
        where  name = 'distribuidora'
         and   type = 'U')
  drop table distribuidora
go

if exists (select 1
         from  sysobjects
        where  name = 'ConfirmaLlamada'
         and   type = 'U')
  drop table ConfirmaLlamada
go

if exists (select 1
         from  sysobjects
        where  name = 'tasalider'
         and   type = 'U')
  drop table tasalider
go

if exists (select 1
         from  sysobjects
        where  name = 'registrallamada'
         and   type = 'U')
```

```
  drop table registrallamada
go

if exists (select 1
          from  sysobjects
         where  name = 'tipo_producto'
          and   type = 'U')
  drop table tipo_producto
go

if exists (select 1
          from  sysobjects
         where  name = 'producto'
          and   type = 'U')
  drop table producto
go

if exists (select 1
          from  sysobjects
         where  name = 'marcas'
          and   type = 'U')
  drop table marcas
go

if exists (select 1
          from  sysobjects
         where  name = 'tipo_esquema'
          and   type = 'U')
  drop table tipo_esquema
go

if exists (select 1
          from  sysobjects
         where  name = 'tipo_seguros'
          and   type = 'U')
  drop table tipo_seguros
go

if exists (select 1
          from  sysobjects
         where  name = 'detalle'
          and   type = 'U')
  drop table detalle
go

if exists (select 1
          from  sysobjects
         where  name = 'tipo_planes'
          and   type = 'U')
  drop table tipo_planes
go
```

```
if exists (select 1
          from  sysobjects
          where  name = 'importe_seguros'
          and   type = 'U')
   drop table importe_seguros
go

if exists (select 1
          from  sysobjects
          where  name = 'plazo'
          and   type = 'U')
   drop table plazo
go

if exists (select 1
          from  sysobjects
          where  name = 'autorizaciones'
          and   type = 'U')
   drop table autorizaciones
go

if exists (select 1
          from  sysobjects
          where  name = 'seguros'
          and   type = 'U')
   drop table seguros
go


/* ============================================================== */
/*   Table : usuarios                                             */
/* ============================================================== */
create table usuarios
(
   user_id              char(8)         not null,
   nombre               char(75)        null,
   aplicación           int             null,
   equipo               int             null,
   codigo_op            char(3)         null
)
go

sp_primarykey usuarios, user_id
go

/* ============================================================== */
/*   Index : usuarios_pk                                          */
/* ============================================================== */
create unique index usuarios_pk on usuarios (user_id)
go
```

```
/* ============================================================ */
/*   Table : tipollamada                                        */
/* ============================================================ */
create table tipollamada
(
   cve_llamada          int             not null,
   descripcion          char(80)        null,
   acceso               char(3)         null
)
go

sp_primarykey tipollamada, cve_llamada
go

/* ============================================================ */
/*   Index : tipollamada_pk                                     */
/* ============================================================ */
create unique index tipollamada_pk on tipollamada (cve_llamada)
go

/* ============================================================ */
/*   Table : publicidad                                         */
/* ============================================================ */
create table publicidad
(
   cve_publicidad       int             not null,
   descripcion_publicidad  char(80)     null
)
go

sp_primarykey publicidad, cve_publicidad
go

/* ============================================================ */
/*   Index : publicidad_pk                                      */
/* ============================================================ */
create unique index publicidad_pk on publicidad (cve_publicidad)
go


/* ============================================================ */
/*   Table : generales                                          */
/* ============================================================ */
create table generales
(
   id_gral              int             not null,
   nombre               char(100)       not null,
   apellido_paterno     char(30)        null,
   apellido_materno     char(30)        null,
   calle_num            char(50)        null,
```

```
   delegacion              char(50)        null,
   colonia                 char(50)        null,
   cve_estado              char(3)         null,
   cve_pais                char(3)         null,
   cve_gral                int             not null
)
go

sp_primarykey generales, id_gral
go

/* ============================================================== */
/*   Index : generales_pk                                         */
/* ============================================================== */
create unique index generales_pk on generales (id_gral)
go

/* ============================================================== */
/*   Index : generales_fk1                                        */
/* ============================================================== */
create index generales_fk1 on generales (cve_gral)
go


/* ============================================================== */
/*   Table : tipo_general                                         */
/* ============================================================== */
create table tipo_general
(
   cve_gral             int             not null,
   descripcion          char(50)        not null

)
go

sp_primarykey tipo_general, cve_gral
go

/* ============================================================== */
/*   Index : tipo_general_pk                                      */
/* ============================================================== */
create unique index tipo_general_pk on tipo_general (cve_gral)
go

/* ============================================================== */
/*   Table : telefonos                                            */
/* ============================================================== */
create table telefonos
(
   id_gral                 int             not null,
   cve_tipo                int             not null,
```

```
   numero_telefonico          char(10)        not null
)
go

sp_primarykey telefonos, id_gral, cve_tipo
go

/* ============================================================== */
/*  Index : telefonos_pk                                          */
/* ============================================================== */
create unique index telefonos_pk on telefonos (id_gral,cve_tipo)
go

/* ============================================================== */
/*  Index : telefonos_fk1                                         */
/* ============================================================== */
create index telefonos_fk1 on telefonos (id_gral)
go

/* ============================================================== */
/*  Index : telefonos_fk2                                         */
/* ============================================================== */
create index telefonos_fk2 on telefonos (cve_tipo)
go

/* ============================================================== */
/*  Table : tipo_telefonos                                        */
/* ============================================================== */
create table tipo_telefonos
(
   cve_tipo             int             not null,
   descripcion          char(30)        not null

)
go

sp_primarykey tipo_telefonos, cve_tipo
go

/* ============================================================== */
/*  Index : tipo_telefonos_pk                                     */
/* ============================================================== */
create unique index tipo_telefonos_pk on tipo_telefonos (cve_tipo)
go

/* ============================================================== */
/*  Table : registro_contactos                                    */
/* ============================================================== */
create table registro_contactos
(
```

```
    id_gral                 int             not null,
    cve_estatus             int             not null,
    fecha_insercion         char(8)         not null,
    hora_insercion          char(8)         not null,
    user_id                 char(8)         not null,
    fecha_sig_contacto      char(8)         not null,
    hora_sig_contacto       char(8)         not null,
    comentarios             char(150)       null

)
go

sp_primarykey registro_contactos,fecha_insercion,hora_insercion, user_id
go

/* ============================================================== */
/*   Index : registro_contactos_pk                                */
/* ============================================================== */
create unique index registro_contactos_pk on registro_contactos
(fecha_insercion,hora_insercion,user_id)
go

/* ============================================================== */
/*   Index : registro_contactos_fk1                               */
/* ============================================================== */
create index registro_contactos_fk1 on registro_contactos (id_gral)
go

/* ============================================================== */
/*   Index : registro_contactos_fk2                               */
/* ============================================================== */
create index registro_contactos_fk2 on registro_contactos (cve_estatus)
go

/* ============================================================== */
/*   Table : estatus                                              */
/* ============================================================== */
create table estatus
(
    cve_estatus             int             not null,
    descripcion             char(30)        not null
)
go

sp_primarykey estatus,cve_estatus
go

/* ============================================================== */
/*   Index : estatus_pk                                           */
/* ============================================================== */
create unique index estatus_pk on estatus (cve_estatus)
```

```
go
/* ============================================================== */
/*   Table : distribuidora                                        */
/* ============================================================== */
create table distribuidora
(
   id_gral2             int             not null,
   cve_marca            int             not null,
   numeroafiliacion     char(8)         not null
)
go

sp_primarykey distribuidora, id_gral2
go

/* ============================================================== */
/*   Index : distribuidora_pk                                     */
/* ============================================================== */
create unique index distribuidora_pk on distribuidora (id_gral2)
go

/* ============================================================== */
/*   Index : distribuidora_fk1                                    */
/* ============================================================== */
create index distribuidora_fk1 on distribuidora (cve_marca)
go

/* ============================================================== */
/*   Table : ConfirmaLlamada                                      */
/* ============================================================== */
create table ConfirmaLlamada
(
   fecha_co             char(8)         not null,
   hora_co              char(8)         not null,
   folio                char(20)        null,
   numllamada           int             null,
   User_id_co           char(8)         not null
)
go

sp_primarykey ConfirmaLlamada, fecha_co, hora_co,User_id_co
go

/* ============================================================== */
/*   Index : ConfirmaLlamada_pk                                   */
/* ============================================================== */
create unique index ConfirmaLlamada_pk on ConfirmaLlamada (fecha_co,
hora_co,User_id_co)
go

/* ============================================================== */
```

```
/*  Index : ConfirmaLlamada_fk1                    */
/* ============================================================ */
create index ConfirmaLlamada_fk1 on ConfirmaLlamada (folio)
go

/* ============================================================ */
/*  Table : tasalider                              */
/* ============================================================ */
create table tasalider
(
   fecha                char(8)         not null,
   tasa                 float           null
)
go

sp_primarykey tasalider, fecha
go

/* ============================================================ */
/*  Index : tasalider_pk                           */
/* ============================================================ */
create unique index tasalider_pk on tasalider (fecha)
go

/* ============================================================ */
/*  Table : registrallamada                        */
/* ============================================================ */
create table registrallamada
(
   fecha                char(8)         not null,
   hora                 char(8)         not null,
   user_id              char(8)         not null,
   cve_llamada          int             null,
   cve_publicidad       int             null,
   cve_producto         char(2)         null,
   cve_plan             char(3)         null,
   id_cte               int             not null,
   id_eje               int             not null,
   id_dis               int             not null,
   id_suc               int             not null,
   cve_nissan           int             null,
   origen               char(2)         null,
   estado               char(2)         null,
   cve_referencia       char(3)         null,
   cta_referencia       char(20)        null,
   comentarios          char(200)       null
)
go

sp_primarykey registrallamada, fecha, hora,user_id
go
```

```
/* ============================================================ */
/*  Index : registrallamada_pk                                  */
/* ============================================================ */
create unique index registrallamada_pk on registrallamada (fecha, hora,user_id)
go

/* ============================================================ */
/*  Index : registrallamada_fk1                                 */
/* ============================================================ */
create index registrallamada_fk1 on registrallamada (user_id)
go

/* ============================================================ */
/*  Index : registrallamada_fk2                                 */
/* ============================================================ */
create index registrallamada_fk2 on registrallamada (cve_llamada)
go

/* ============================================================ */
/*  Index : registrallamada_fk3                                 */
/* ============================================================ */
create index registrallamada_fk3 on registrallamada (cve_publicidad)
go

/* ============================================================ */
/*  Index : registrallamada_fk4                                 */
/* ============================================================ */
create index registrallamada_fk4 on registrallamada (cve_producto)
go

/* ============================================================ */
/*  Index : registrallamada_fk5                                 */
/* ============================================================ */
create index registrallamada_fk5 on registrallamada (id_cte)
go

/* ============================================================ */
/*  Index : registrallamada_fk6                                 */
/* ============================================================ */
create index registrallamada_fk6 on registrallamada (id_eje)
go

/* ============================================================ */
/*  Index : registrallamada_fk7                                 */
/* ============================================================ */
create index registrallamada_fk7 on registrallamada (id_dis)
go

/* ============================================================ */
/*  Index : registrallamada_fk8                                 */
```

```
/* ============================================================== */
create index registrallamada_fk8 on registrallamada (id_suc)
go

/* ============================================================== */
/*  Index : registrallamada_fk9                                   */
/* ============================================================== */
create index registrallamada_fk9 on registrallamada (cve_plan)
go


/* ============================================================== */
/*  Table : tipo_planes                                           */
/* ============================================================== */
create table tipo_planes
(
   cve_plan             char(3)         not null,
   descripcion          char(50)        null,
   cve_producto         char(2)         null
)
go

sp_primarykey tipo_planes, cve_plan
go

/* ============================================================== */
/*  Index : tipo_planes_pk                                        */
/* ============================================================== */
create unique index tipo_planes_pk on tipo_planes (cve_plan)
go

/* ============================================================== */
/*  Table : detalle                                               */
/* ============================================================== */
create table detalle
(
   cve_esquema          char(2)         not null,
   cve_plan             char(3)         not null,
   cve_producto         char(2)         not null,
   cve_seguros          int             not null,
   persona              char(2)         not null,
   tipo_ns              char(2)         not null,
   tipo_tasa            char(2)         not null,
   plazo                int             not null,
   eng_min              int             null,
   empresa              char(5)         null,
   tasa                 float           null,
   piso                 float           null,
   factor               float           null,
   techo                float           null,
   refinan              float           null,
```

```
   eng_max                 float           null
)
go

sp_primarykey detalle,
cve_esquema,cve_plan,cve_producto,cve_seguros,persona,tipo_ns,tipo_tasa,plazo
go

/* ============================================================== */
/*   Index : detalle_pk                                           */
/* ============================================================== */
create unique index detalle_pk on detalle (cve_esquema, cve_plan, cve_producto,
cve_seguros, persona,tipo_ns, tipo_tasa,plazo)
go

/*============================================================== */
/*   Index : detalle_fk1                                          */
/* ============================================================== */
create index detalle_fk1 on detalle (cve_esquema)
go

/* ============================================================== */
/*   Index : detalle_fk2                                          */
/* ============================================================== */
create index detalle_fk2 on detalle (cve_plan)
go

/* ============================================================== */
/*   Index : detalle_fk3                                          */
/* ============================================================== */
create index detalle_fk3 on detalle (cve_producto)
go

/* ============================================================== */
/*   Index : detalle_fk4                                          */
/* ============================================================== */
create index detalle_fk4 on detalle (cve_seguros)
go

/* ============================================================== */
/*   Table : tipo_seguros                                         */
/* ============================================================== */
create table tipo_seguros
(
   cve_seguros             int             not null,
   descripcion             char(50)        null
)
go

sp_primarykey tipo_seguros, cve_seguros
go
```

```
/* ============================================================ */
/*   Index : tipo_seguros_pk                                    */
/* ============================================================ */
create unique index tipo_seguros_pk on tipo_seguros (cve_seguros)
go


/* ============================================================ */
/*   Table : tipo_esquema                                       */
/* ============================================================ */
create table tipo_esquema
(
   cve_esquema          char(2)         not null,
   descripcion          char(50)        null
)
go

sp_primarykey tipo_esquema, cve_esquema
go

/* ============================================================ */
/*   Index : tipo_esquema_pk                                    */
/* ============================================================ */
create unique index tipo_esquema_pk on tipo_esquema (cve_esquema)
go

/* ============================================================ */
/*   Table : marcas                                             */
/* ============================================================ */
create table marcas
(
   cve_marca            int             not null,
   descripcion          char(30)        null
)
go

sp_primarykey marcas, cve_marca
go

/* ============================================================ */
/*   Table : producto                                           */
/* ============================================================ */
create table producto
(
   cve_bienproducto     char(7)         not null,
   modelo               int             not null,
   descripcion          char(80)        null,
   precio_min           float           null,
   precio_max           float           null,
   cve_producto         char(2)         null,
```

```
   cve_marca                int              null
)
go

sp_primarykey producto, cve_bienproducto, modelo
go

/* ============================================================== */
/*   Index : producto_pk                                          */
/* ============================================================== */
create unique index producto_pk on producto (cve_bienproducto, modelo)
go

/* ============================================================== */
/*   Index : producto_fk1                                         */
/* ============================================================== */
create index producto_fk1 on producto (cve_producto)
go

/* ============================================================== */
/*   Index : producto_fk2                                         */
/* ============================================================== */
create index producto_fk2 on producto (cve_marca)
go


/* ============================================================== */
/*   Table : tipo_producto                                        */
/* ============================================================== */
create table tipo_producto
(
   cve_producto          char(2)        not null,
   descripcion           char(50)       null
)
go

sp_primarykey tipo_producto, cve_producto
go

/* ============================================================== */
/*   Index : tipo_producto_pk                                     */
/* ============================================================== */
create unique index tipo_producto_pk on tipo_producto (cve_producto)
go

/* ============================================================== */
/*   Table : importe_seguros                                      */
/* ============================================================== */
create table importe_seguros
(
   cve_bienproducto      char(7)        not null,
```

```
    modelo                  int             not null,
    cve_plazo               int             not null,
    importe                 float           null

)
go

sp_primarykey importe_seguros, cve_bienproducto, modelo, cve_plazo
go

/* ========================================================== */
/*   Index : importe_seguros_pk                               */
/* ========================================================== */
create unique index importe_seguros_pk on importe_seguros
(cve_bienproducto,modelo,cve_plazo)
go

/* ========================================================== */
/*   Table : plazo                                            */
/* ========================================================== */
create table plazo
(
    cve_plazo           int             not null,
    descripcion         char(20)        null

)
go

sp_primarykey plazo, cve_plazo
go

/* ========================================================== */
/*   Index : cve_plazo_pk                                     */
/* ========================================================== */
create unique index plazo_pk on plazo (cve_plazo)
go

/* ========================================================== */
/*   Table : autorizaciones                                   */
/* ========================================================== */
create table autorizaciones
(
    distribuidora           char(80)            not null,
    marca                   char(30)            not null,
    modelo                  int                 not null,
    cve_producto            char(3)             not null,
    plazo                   char(3)             not null,
    tipo_cta                char(3)             not null,
    importe                 char(20)            not null,
    num_cta                 char(20)            not null,
    num_prestamo            char(20)            not null,
```

```
    num_autorizaciones      char(20)            not null,
    tipo_plan               char(80)            not null,
    fehca                   char(8)             not null,
    hora                    char(8)             not null,
    nombre_cte              char(80)            not null,
    usuario                 char(8)             not null
)
go

sp_primarykey autorizaciones, tipo_cta, fehca, hora,usuario
go

/* ============================================================== */
/*  Index : autorizaciones_pk                                     */
/* ============================================================== */
create unique index autorizaciones_pk on autorizaciones (tipo_cta, fehca, hora, usuario)
go


/* ============================================================== */
/*  Table : seguros                                               */
/* ============================================================== */
create table seguros
(
        cve_seguros         int                 not null,
        tipo_ns             char(2)             not null,
        cve_producto        char(2)             not null,
        cve_enviar          char(1)             null
)
go

sp_primarykey seguros, cve_seguros,tipo_ns,cve_producto
go

sp_foreignkey registrallamada, usuarios, user_id
go

sp_foreignkey registrallamada, tipollamada, cve_llamada
go

sp_foreignkey registrallamada, publicidad, cve_publicidad
go

sp_foreignkey registrallamada, tipo_producto, cve_producto
go

sp_foreignkey registrallamada, generales, id_cte
go

sp_foreignkey registrallamada, generales, id_eje
go
```

```
sp_foreignkey registrallamada, generales, id_suc
go

sp_foreignkey registrallamada, generales, id_dis
go

sp_foreignkey generales, tipo_general, cve_gral
go

sp_foreignkey telefonos, generales, id_gral
go

sp_foreignkey telefonos, tipo_telefonos, cve_tipo
go

sp_foreignkey registro_contactos, generales, id_gral
go

sp_foreignkey registro_contactos, estatus, cve_estatus
go

sp_foreignkey distribuidora, marcas, cve_marca
go

sp_foreignkey detalle, tipo_esquema, cve_esquema
go

sp_foreignkey detalle, tipo_planes, cve_plan
go

sp_foreignkey detalle, tipo_producto, cve_producto
go

sp_foreignkey detalle, tipo_seguros, cve_seguros
go

sp_foreignkey producto, tipo_producto, cve_producto
go

sp_foreignkey producto, marcas, cve_marca
go

sp_foreignkey importe_seguros, producto, cve_bienproducto,modelo
go

sp_foreignkey importe_seguros, plazo, cve_plazo
go
```