UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES
ARAGÓN

ñINGENIERÍA INVERSA DE BASE DE DATOSò

TESIS

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

P R E S E N T A:

CLAUDIA ALEJANDRA CORONADO PASTOR

ASESOR:
MAT. LUIS RAMÍREZ FLORES

ESTADO DE MÉXICO, 2008

Dedicatoria

A mi Madre, porque su amor ha sido la más grande bendición de mi vida.

A mi Padre, por ser su ñ*hermoso cariño*ò

A mi Hermano, porque me ha mostrado que se puede salir adelante.

A Mateo, hasta siempre amigo mío.

A mis amigos, Eli, Elena, Ángel, Adrian, Alejandro, César, Ernesto, José, Luis, Marco, Manuel por estar conmigo en los buenos y malos momentos.

ñ*Caer está permitido, levantarse es una obligación***ò** Proverbio ruso

Agradecimientos

La presente tesis, si bien requirió esfuerzo y dedicación de mi parte, no hubiera podido realizarse sin la aportación desinteresada de cada una de las personas que a continuación mencionaré con enorme agradecimiento.

A mi asesor, Mat. Luis Ramírez, por la disposición y confianza mostrada en todo momento a lo largo de estos años de trabajo, así como su orientación para la realización de esta tesis.

Al Ing. Roberto Blanco, por sus sugerencias y consejos para el mejoramiento de este trabajo.

Al Ing. Manuel Quintero, por las observaciones y comentarios que ayudaron a alcanzar los objetivos de esta tesis.

Al Ing. Oscar Estrada quien contribuyó con notables propuestas y recomendaciones acertadas.

Al M. en C. Marcelo Pérez, por su apoyo, dirección y valioso tiempo brindado.

Al Ing. Roberto Márquez por su cooperación y paciencia.

A la Ing. Carmen Riva Palacio y al Ing. Gregorio Matadamas por las facilidades que me proporcionaron para que mis responsabilidades laborales no interfirieran en la realización de esta meta.

A la UNAM y sus profesores que me formaron durante mis años de estudio.

Índice

Índice de Figuras

Índice de Ilustraciones

Introducción

Las Bases de Datos almacenan elementos informáticos, como nombres de personas o direcciones. El propósito principal del almacenamiento de datos es la recuperación de éstos en cualquier momento y así poder tomar decisiones o realizar acciones.

El objetivo fundamental del presente trabajo consiste en encontrar una estrategia de solución al problema que las organizaciones pueden llegar a presentar cuando el manejo de su información, por medio de Base de Datos, resulta inconveniente, ya sea por un mal diseño o la poca documentación que se cuenta para la explotación de la misma. Tomando en cuenta la premisa de que la información contenida en una Base de Datos es valiosa para una empresa, en esta tesis se expone una alternativa para que dicha Base pueda optimizarse.

La Ingeniería Inversa es dicha alternativa y al final de esta tesis, se podrá concluir que tan viable puede ser esta solución.

Hace algunos años, las empresas creaban sus propios programas para manejar la información, pero esto resultaba caótico, inconsistente y redundante y no podían interactuar con otros sistemas. De esta forma se han ido desarrollando sistemas que gestionan la información de una Base de Datos y que contemplan, además, mecanismos de seguridad y recuperación de datos. Así, cualquier lenguaje puede interactuar con estos sistemas y las empresas pueden crear programas para que la información sea mostrada a los usuarios de forma conveniente.

En el capítulo I, Introducción a las Bases de Datos, se da una explicación de las características de una Base de Datos, como son, los modelos de datos, el lenguaje utilizado para el manejo de Datos, los sistemas que las gestionan, la arquitectura, ventajas e inconvenientes de las Bases de Datos.

En el capítulo II, Ingeniería Inversa de Base de Datos, se presenta una introducción a la evolución de los sistemas informáticos, que puede llegar a considerar la reingeniería como una forma de mejorar la operación y funcionalidad, misma que se apoyará en la Ingeniería Inversa. También se da la descripción del método genérico para la aplicación de Ingeniería Inversa sobre las Bases de Datos.

En el capítulo III, Herramientas de Ingeniería Inversa de Base de Datos, se presenta las Herramientas CASE, que son un gran soporte para los sistemas de información y que han evolucionado para asistir a la Ingeniería Inversa.

En el capítulo IV, Seguridad de Base de Datos, se da una breve explicación de las medidas de seguridad sobre las Bases de Datos, no solo sobre la información, sino en el aspecto jurídico ya que la Ingeniería Inversa de Base de Datos trabaja sobre los datos y la estructura de una Base anterior.

Introducción a las Bases de Datos

La gran necesidad de poder obtener el máximo control sobre la información, siendo está de cualquier índole, ha llevado al hombre al desarrollo de las Bases de Datos en una de las actividades más importantes en el campo de la Informática. De hecho, el poder de gestionar grandes cantidades de datos se ha constituido, especialmente a través de las últimas décadas, en uno de los factores más significativos en lo que respecta al nivel de progreso del hombre.

Es por ello que se han creado sistemas encargados de realizar este tipo de gestión automática de la información, sin tener por ello que temer en ningún momento las consecuencias derivadas de dicho control.

El avance de la computación ha favorecido en gran medida a la realización de éste objetivo, puesto que se ha ideado para encargarse de realizar todas aquellas operaciones que al hombre le costaban un gran esfuerzo y cantidad de tiempo. Operaciones de carácter repetitivo en las que antes el ser humano debía emplear horas, como puede ser la captura y gestión de nombres de personas o sus direcciones domiciliarias, en nuestros días a una computadora no le lleva más de unos segundos. Conforme mejora la calidad y rendimiento de los equipos informáticos, mayor capacidad de cálculo son capaces de ofrecer, por lo que, incluso la organización o búsqueda de información archivada no representa ningún problema.

El desarrollo de las Bases de Datos han dado como resultado dos procedimientos importantes para una empresa: el procesamiento de transacciones[1] y el DataWareHousing[2]. Pero no solo las necesidades de gestión del área empresarial han obtenido beneficios, las Bases de Datos cubren también las necesidades domésticas del proceso de información.

1.1 Historia de los Sistemas de Base de Datos

Una Base de Datos se define como la colección de datos organizados bajo normas establecidas en un modelo seleccionado.

La Base de Datos es un gran conjunto de datos que se define una sola vez y que se utiliza al mismo tiempo por muchos usuarios. En lugar de trabajar con archivos desconectados e información redundante, todos los datos se integran con una mínima cantidad de duplicidad. La Base de Datos no sólo contiene los datos, también almacena una descripción de dichos datos. Esta descripción es lo que se denomina metadatos[3], se almacena en el diccionario de datos[4] o catálogo y es lo que permite que exista independencia de datos lógica y física.

[1] La tecnología responsable de que el intercambio se realice de forma equilibrada y predecible se denomina procesamiento de transacciones. Las transacciones garantizan que los recursos orientados a datos no se actualicen permanentemente salvo que todas las operaciones de la unidad transaccional se completen de forma satisfactoria.
[2] Un Data Warehouse es una colección de datos en la cual se encuentra integrada la información de una Institución y que se usa como soporte para el proceso de toma de decisiones gerenciales.
[3] Los metadatos guardan información sobre los formatos, significado y origen de los datos y facilitan, por lo tanto, el acceso, la navegación y la administración de los datos.
[4] Contiene la información referente a la estructura de la Base de Datos

Los sistemas de Bases de Datos tienen sus raíces en el proyecto estadounidense Apolo en los años sesenta. En aquella época, no había ningún sistema que permitiera gestionar la inmensa cantidad de información que requería el proyecto. La primera empresa encargada del proyecto, North American Aviation (Navegación Aérea de Norteamérica o NNA), desarrolló un software denominado General Update Access Method (Método de acceso de actualización general o GUAM) que estaba basado en el concepto de que varias piezas pequeñas se unen para formar una pieza más grande, y así sucesivamente hasta que el producto final está ensamblado. Esta estructura, que tiene la forma de un árbol, es lo que se denomina una estructura jerárquica. A mediados de los sesenta, IBM (Internacional Business Machines Corporation) se unió a NAA para desarrollar GUAM en lo que ahora se conoce como Information Management System (Sistema manejador de información o IMS). El motivo por el cual IBM restringió IMS al manejo de jerarquías de registros fue el de permitir el uso de dispositivos de almacenamiento serie, más exactamente las cintas magnéticas, ya que era un requisito del mercado por aquella época.

A mitad de los sesenta, se desarrolló Integrated Data Store(Almacenamiento de datos integrados o IDS), de General Electric. Este trabajo fue dirigido por uno de los pioneros en los sistemas de Bases de Datos, Charles Bachmann. IDS era un nuevo tipo de sistema de Bases de Datos conocido como sistema de red, que produjo un gran efecto sobre los sistemas de información de aquella generación. El sistema de red se desarrolló, en parte, para satisfacer la necesidad de representar relaciones entre datos más complejos que las que se podían modelar con los sistemas jerárquicos, y, en parte, para imponer un estándar de Bases de Datos. Para ayudar a establecer dicho estándar, el Congreso de sistemas de lenguajes de datos, formado por representantes del Gobierno de Estados Unidos y representantes del mundo empresarial, formaron un grupo denominado DBTG (Data Base Task Group o Grupo de Trabajo sobre Base de Datos), cuyo objetivo era definir unas especificaciones estándar que permitieran la creación de Bases de Datos y el manejo de los datos. El DBTG presentó su informe final en 1971 y aunque éste no fue formalmente aceptado por American National Standards Institute (Instituto Nacional Americano de estándares o ANSI), muchos sistemas se desarrollaron siguiendo la propuesta del DBTG. Estos sistemas son los que se conocen como sistemas de red, o sistemas CODASYL o DBTG.

En 1970 Dr. Edgar Frank Codd, de los laboratorios de investigación de IBM, escribió un artículo presentando el modelo relacional. En este artículo, presentaba también los inconvenientes de los sistemas previos, el jerárquico y el de red. Entonces, se comenzaron a desarrollar numerosos sistemas relacionales, apareciendo los primeros a finales de los setenta y principios de los ochenta. Uno de los primeros es System R, de IBM, que se desarrolló para probar la funcionalidad del modelo relacional, proporcionando una implementación de sus estructuras de datos y sus operaciones.

La propia evolución de la Informática, en todos sus aspectos, fue reduciendo costos y simplificando procesos. Las herramientas de administración de Bases de

Datos se fueron desarrollando hasta convertirse en poderosos motores capaces de manejar enormes cantidades de información en pocos segundos.

A medida que la necesidad de almacenar y gestionar la información fue creciendo, la Informática ha procurado diferentes soluciones mediante la definición de una serie de modelos. Estos modelos son un conjunto de pautas, mediante las cuales podemos representar una realidad de forma abstracta mediante datos y la forma de estructurar éstos.

1.2 Modelo de datos

Los requerimientos de información para un área funcional o para toda una organización se puede representar en forma lógica mediante un modelo de datos. Jeffrey Ullman define que un modelo de datos es un sistema formal y abstracto que permite describir los datos de acuerdo con reglas y convenios predefinidos. Es formal, pues los objetos del sistema se manipulan siguiendo reglas perfectamente definidas y utilizando exclusivamente los operadores definidos en el sistema, independientemente de lo que estos objetos y operadores puedan significar.

El Modelo de Datos permite aprender parcialmente del mundo, ya que es probable que no se pueda obtener un modelo que capture en su totalidad la realidad. Sin embargo, con esta herramienta se puede adquirir un conocimiento relativamente completo de cualquier área de estudio. Así, un modelo captura tanto conocimiento como sea necesario para cumplir con los requerimientos establecidos previamente.

Un modelo de datos tiene tres componentes básicos que son:

1. Estructuras de datos. Es la colección de objetos abstractos formados por los datos.
2. Operadores entre las estructura. Es el conjunto de operadores con reglas bien definidas que permiten manipular las estructuras de datos.
3. Definiciones de integridad. Es una colección de conceptos y reglas que permite expresar que valores son válidos para los datos en el modelo.

Dentro de las características primordiales de un Modelo de Datos están las siguientes:

- Sencillo: Como regla general, los atributos que comprenden una entidad cualquiera deberían describir sólo a dicha entidad.
- Sin redundancias: Ningún atributo, excepto las claves, debe describir a más de una entidad.
- Flexible y adaptable ante necesidades futuras: En la mayoría de los casos se construyen aplicaciones pensando solamente en las necesidades actuales, esto da origen a la reescritura de cientos de líneas de código cuando se necesita introducir un nuevo requerimiento de información.

- Facilidad para integrarse con otros modelos de datos: Los detalles de integración con otras aplicaciones deben ser tomados en cuenta para procurar que los diseños de Modelo de Datos sean lo más independientes de los aspectos físicos (plataforma de aplicaciones, tecnología, etc.) y de las limitaciones propias que podría plantear una aplicación específica.

Se dividen en tres grupos, modelos lógicos basados en registros, modelos lógicos basados en objetos y modelos físicos de datos.

1.2.1 Modelos lógicos basados en registros

Estos modelos utilizan registros e instancias para representar la realidad, así como las relaciones que existen entre estos registros. Se usan para especificar la estructura lógica global de la Base de Datos y para proporcionar una descripción a nivel más alto de la implementación.

Los tres modelos de datos más conocidos son:

1.2.1.1 Modelo Jerárquico

Este modelo representa la información a través de una estructura de árboles. Un registro se divide en segmentos que se conectan entre sí en relaciones padres e hijos. La relación entre sus registros consiste en que uno se relacione con otro, o bien que uno solo se relaciones con muchos. Como se mencionó anteriormente, este modelo fue el primero en aparecer y se caracteriza por organizar la información a través de una estructura de árboles.

En la actualidad ya no es usado por la gran cantidad de inconvenientes. La flexibilidad y falta de estructuración de este modelo permite crear registros cuyos campos sean variables en número y tamaño; además cuando la información almacenada en la Base de Datos es muy grande, se convierte en inmanejable. En la figura 1-1 se muestra un ejemplo de este modelo.

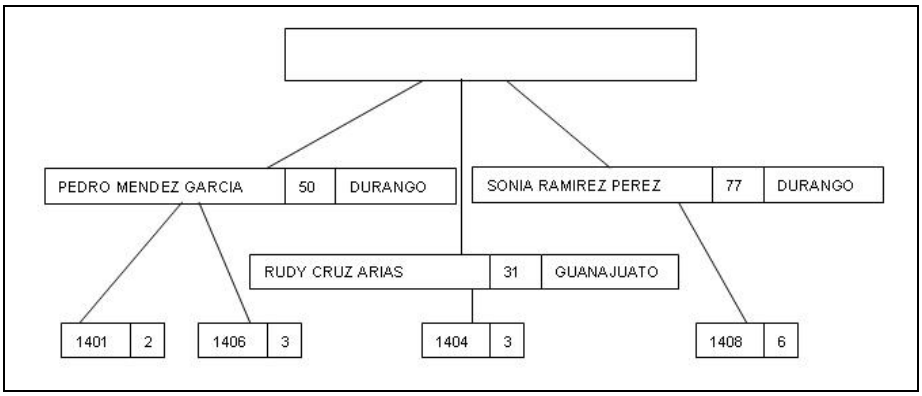


Figura 0-1 Modelo jerárquico

1.2.1.2 Modelo de Red

Este modelo, mostrado en la figura 1-2, fue concebido como una ampliación del modelo jerárquico, cuya finalidad era solucionar las deficiencias lógicas de este último. Al igual que el anterior, también se emplea un árbol como estructura base, pero con la diferencia de que un mismo hijo puede tener diferentes padres, con lo que es posible representar relaciones muchos a muchos sin redundancia aparente.

El modelo en red tiene un carácter totalmente general. En el modelo en red no se especifica ningún tipo de restricción en lo que respecta a las interrelaciones. Esto quizá haga del modelo en red un modelo sencillo de utilizar, pero no deja de tener un carácter general y provoca que en la práctica su uso no resulte nada fácil.

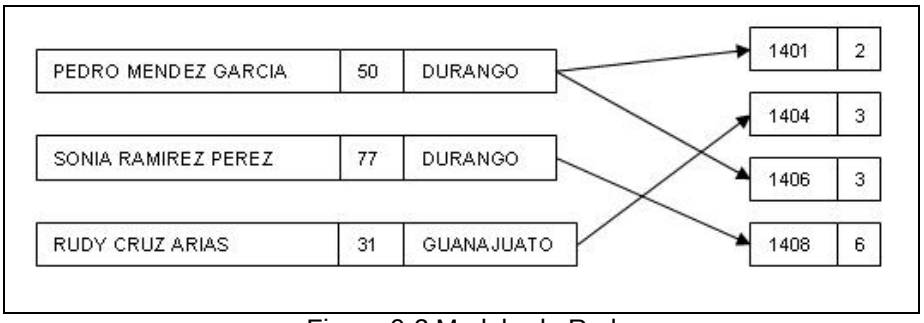


Figura 0-2 Modelo de Red

1.2.1.3 Modelo relacional

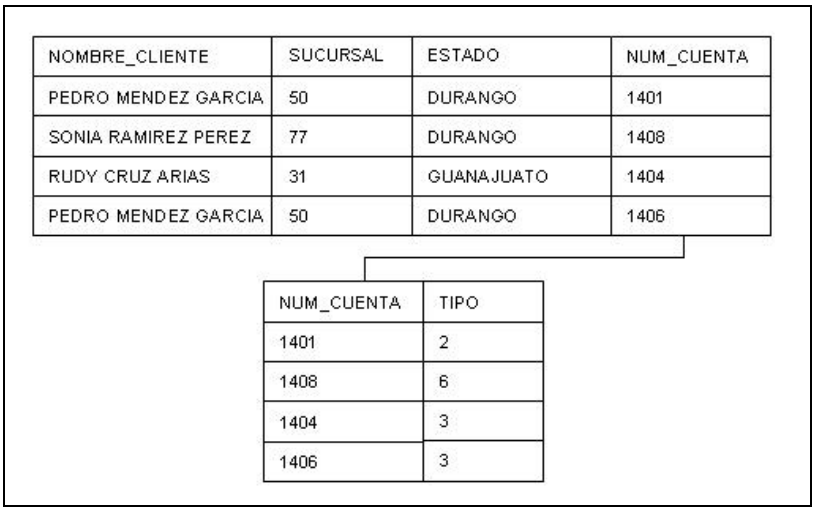
El Modelo Relacional, mostrado en la figura 1-3, es una de las técnicas más utilizadas debida a que su fundamento matemático denominado Álgebra Relacional[5] ha facilitado su aplicación en la construcción de Bases de Datos relacionales permitiendo la creación de muchos manipuladores de Bases de Datos de gran difusión comercial.

Codd definió tres características del modelo relacional:

1. Son estructuras de datos simples. Consiste en tablas de dos dimensiones donde los elementos son ítem de datos. Esto permite un alto grado de independencia de la representación física de los datos.
2. El modelo relacional provee la consistencia de los datos. El diseño de las Bases de Datos es asistido por los procesos de normalización que elimina las anomalías en los datos. Adicionalmente, los estados de consistencia de la Base de Datos puede ser uniformemente definida y mantenida a través de reglas de integridad.

[5] Las operaciones de álgebra relacional manipulan relaciones, es decir , estas operaciones usan uno o dos relaciones existentes para crear una nueva relación

3. El modelo relacional permite la manipulación de las relaciones. Esta característica puede ser encargada a potentes lenguajes basados en la teoría (Álgebra Relacional) o en la lógica (Cálculo Relacional[6]).



NOMBRE_CLIENTE	SUCURSAL	ESTADO	NUM_CUENTA
PEDRO MENDEZ GARCIA	50	DURANGO	1401
SONIA RAMIREZ PEREZ	77	DURANGO	1408
RUDY CRUZ ARIAS	31	GUANAJUATO	1404
PEDRO MENDEZ GARCIA	50	DURANGO	1406

NUM_CUENTA	TIPO
1401	2
1408	6
1404	3
1406	3

Figura 0-3 Modelo Relacional

En este modelo se representan los datos y las relaciones entre estos. La estructura de datos relacional tiene como elemento fundamental la relación. Una relación constituye lo que se puede llamar una tabla. Una tupla corresponde a una fila de esta tabla y un atributo o característica a una columna. El número de tuplas de una relación se denomina cardinalidad y el número de atributos se denomina grado (Figura 1-4).

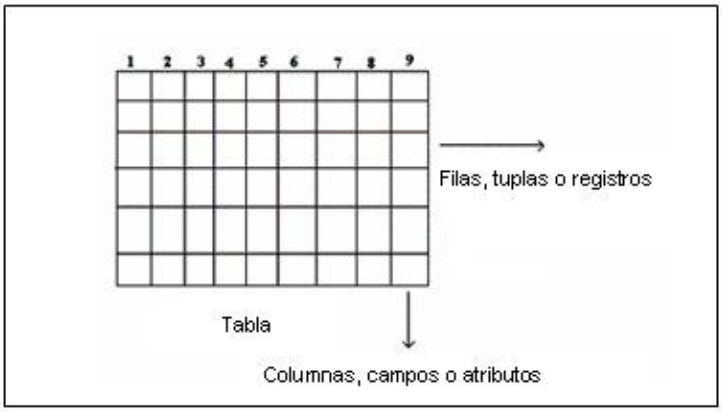


Figura 0-4 Tablas

Una tabla es una colección de objetos que tienen los mismos atributos, y debe cumplir con ciertas restricciones:

- Las celdas de la tabla deben tener un solo valor, no se permiten arreglos o conjuntos de valores.

[6] El cálculo relacional es basado en la rama de la lógica matemática llamado cálculo de predicados.

- Todos los valores de cualquier columna deben ser del mismo tipo.
- Cada atributo o columna tiene un nombre único; por lo tanto, no pueden existir en una misma tabla dos atributos con el mismo nombre.
- El orden de las columnas es insignificante.
- Dos renglones o tuplas de una tabla no pueden ser idénticos.
- El orden de los renglones es insignificante.
- Todas las relaciones deben tener al menos una llave. Una llave o clave es uno o más atributos que identifican a una tupla o renglón de una manera única. En caso extremo, la llave consistiría de todos los atributos de la relación.

Cuando una llave identifica de manera exclusiva a una tupla en una tabla se le denomina llave primaria o criterio primario. Una llave puede denominarse llave secundaria o criterio secundario si no identifica de manera única a una tupla. Tanto la llave primaria como las secundarias se eligen de entre las llaves candidatas[7]. Por otro lado, se denominan claves ajenas o llaves foráneas a una referencia de una relación a otra, mediante su clave. También conocida como una relación implícita.

1.2.2 Modelos lógicos basados en objetos

Un problema de la vida real maneja concepciones abstractas o concretas, tangibles o intangibles, a las cuales se les ha dado el nombre de "objetos". Este modelo representa estos objetos como los percibimos en el mundo real y tienen la capacidad de estructuración bastante flexible y permiten especificar restricciones de datos explícitamente.

1.2.2.1 Modelo Entidad- Relación

Se basa en la percepción de un mundo compuesto por objetos (entidades) con características propias (atributos) y de asociaciones entre esos objetos (relaciones). Su representación grafica se muestra en la figura 1-5.

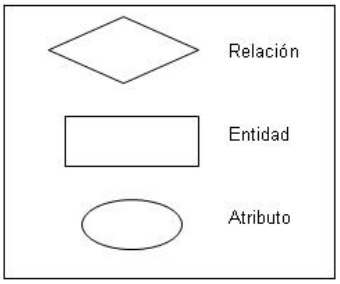


Figura 0-5 Elementos del modelo Entidad-Relación

[7] Es un atributo o conjunto de atributos que identifican unívocamente a una tupla dentro de una relación.

El modelo Entidad-Relación es el modelo conceptual más utilizado para el diseño conceptual de Bases de Datos. Fue introducido por Peter Chen en 1976. El modelo Entidad-Relación está formado por un conjunto de conceptos que permiten describir la realidad mediante un conjunto de representaciones gráficas y lingüísticas, como ejemplo, la figura 1-6.

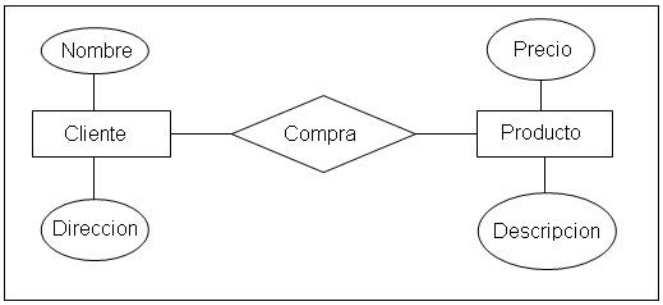


Figura 0-6 Ejemplo del modelo Entidad-Relación

1.2.2.2 Entidad

Es cualquier ente real o abstracto, sobre el que se desea almacenar datos. Las entidades se representan gráficamente mediante rectángulos y su nombre aparece en el interior. Un nombre de entidad sólo puede aparecer una vez en el esquema conceptual.

1.2.2.3 Atributo

Es una característica de interés o un hecho sobre una entidad o sobre una relación. Los atributos representan las propiedades básicas de las entidades y de las relaciones. Toda la información extensiva es portada por los atributos. Gráficamente, se representan mediante círculos pequeños que surgen de las entidades o relaciones a las que pertenecen.

Cada atributo tiene un conjunto de valores asociados denominado dominio. El dominio define todos los valores posibles que puede tomar un atributo. Puede haber varios atributos definidos sobre un mismo dominio.

Los atributos pueden ser simples o compuestos. Un atributo simple es un atributo que tiene un solo componente, que no se puede dividir en partes más pequeñas que tengan un significado propio. Un atributo compuesto es un atributo con varios componentes, cada uno con un significado por sí mismo. Un grupo de atributos se representa mediante un atributo compuesto cuando tienen afinidad en cuanto a su significado, o en cuanto a su uso. Un atributo compuesto se representa gráficamente mediante un óvalo.

Los atributos también pueden clasificarse en monovalentes o polivalentes. Un atributo monovalente es aquel que tiene un sólo valor para cada ocurrencia (datos almacenados en la Base de Datos) de la entidad o relación a la que pertenece. Un atributo polivalente es aquel que tiene varios valores para cada ocurrencia de la

entidad o relación a la que pertenece. A estos atributos también se les denomina multivaluados, y pueden tener un número máximo y un número mínimo de valores.

1.2.2.4 Cardinalidad

La cardinalidad con la que una entidad participa en una relación especifica el número mínimo y el número máximo de correspondencias en las que puede tomar parte cada ocurrencia de dicha entidad. Las reglas que definen la cardinalidad de las relaciones son las reglas de negocio[8]. El valor por omisión es uno a uno.

En otras palabras, indica el número de entidades del conjunto de entidades E2 que se relacionan con una entidad del conjunto de entidades E1 y viceversa.

Dependiendo de esto, puede ser:

- Uno a uno. 1:1. Una entidad del conjunto de entidades E1 se relaciona con una única entidad del conjunto de entidades E2 y viceversa. Véase Figura 1-7.

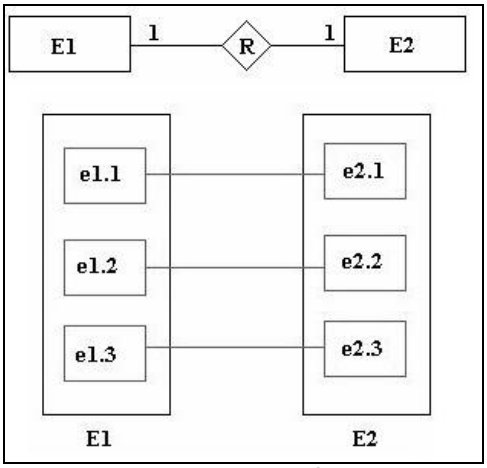


Figura 0-7 Relación 1:1

- Uno a muchos. 1:n. Una entidad del conjunto de entidades E1 se relaciona con muchas entidades del conjunto de entidades E2 y una entidad del conjunto de entidades E2 sólo puede estar relacionada con una entidad del conjunto de entidades E1. Véase Figura 1-8.

[8] Las reglas de negocio son ciertas restricciones específicas que los usuarios o los administradores de la Base de Datos pueden imponer sobre los datos.

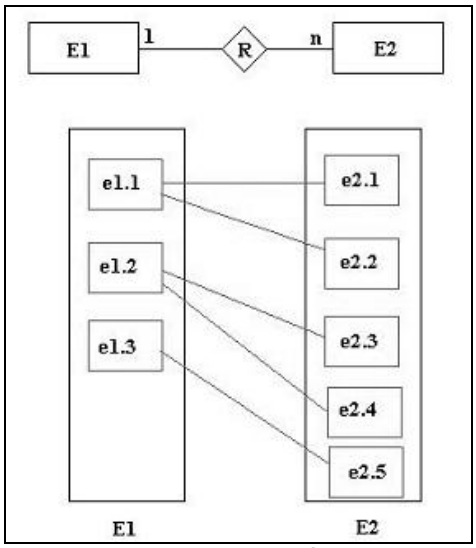


Figura 0-8 Relación 1:n

- Muchos a uno. n:1. Una entidad en E1 está asociada con una única entidad del conjunto de entidades E2 y una entidad del conjunto de entidades en E2 está relacionada con muchas entidades del conjunto de entidades E1. Véase Figura 1-9.

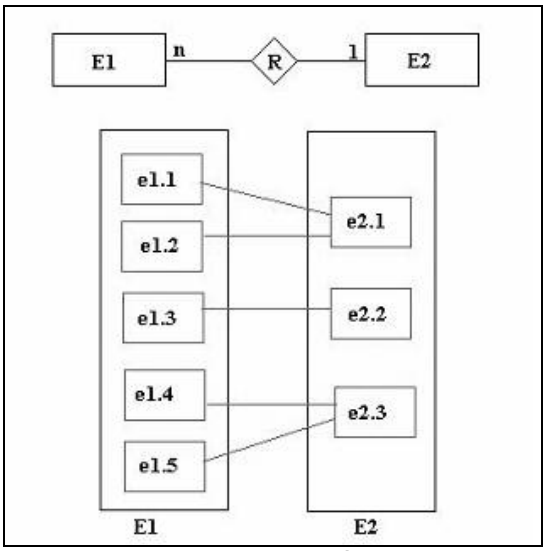


Figura 0-9 Relación n:1

- Muchos a muchos. n:n. Una entidad del conjunto de entidades E1 está relacionada con muchas entidades del conjunto de entidades E2 y viceversa. Véase Figura 1-10.

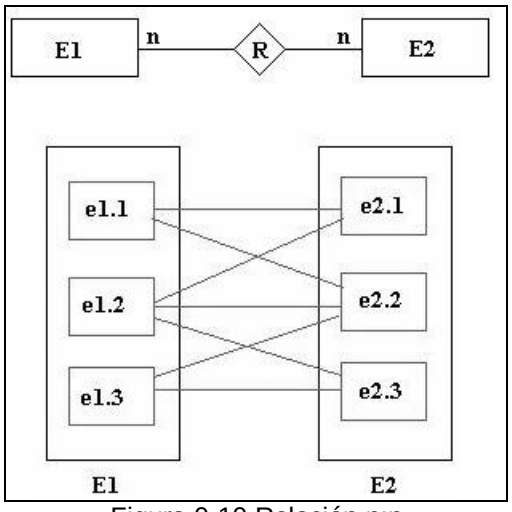


Figura 0-10 Relación n:n

Existen diversas maneras de representar la cardinalidad, la más sencilla es la presentada anteriormente, donde sólo se especifica el número máximo de relaciones que puede tener una entidad con entidades del otro conjunto de entidades con que se asocia.

1.2.2.5 Relación

Es una correspondencia o asociación entre dos o más entidades. Cada relación tiene un nombre que describe su función. Las relaciones se representan gráficamente mediante rombos y su nombre aparece en el interior.

Las entidades que están involucradas en una determinada relación se denominan entidades participantes. El número de participantes en una relación es lo que se denomina grado de la relación. Puede ser llamada:

- Unaria*:* Participa un único conjunto de entidades (Figura 1-11):

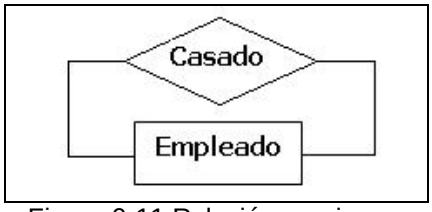


Figura 0-11 Relación unaria

- Binaria: Participan dos conjuntos de entidades (Figura 1-12).

Figura 0-12 Relación binaria

- N-aria*:* Participan más de dos conjuntos de entidades(Figura 1-13) :

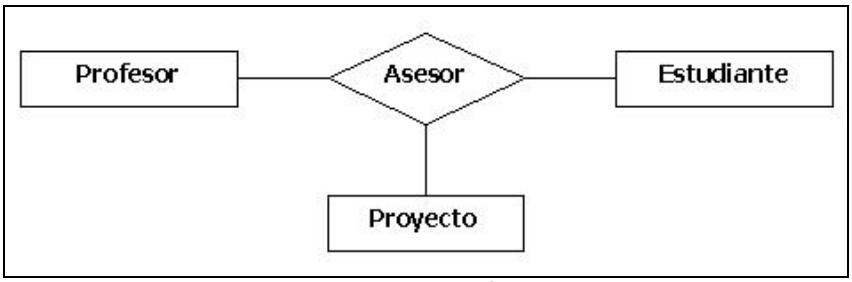


Figura 0-13 Relación n-aria

Una relación recursiva es una relación donde la misma entidad participa más de una vez en la relación con distintos papeles. El nombre de estos papeles es importante para determinar la función de cada participación.

Estas asociaciones pueden determinarse de forma relativamente rápida una vez identificadas las entidades. Existen varios tipos de relaciones. Las más comunes son las relaciones binarias, es decir, relaciones entre dos entidades diferentes.

1.2.2.6 Identificador

Un identificador de una entidad es un atributo o conjunto de atributos que determina de modo único cada ocurrencia de esa entidad. Un identificador de una entidad debe cumplir dos condiciones:

- No pueden existir dos ocurrencias de la entidad con el mismo valor del identificador.
- Si se omite cualquier atributo del identificador, la condición anterior deja de cumplirse.

Toda entidad tiene al menos un identificador y puede tener varios identificadores alternativos. Las relaciones no tienen identificadores.

1.2.2.7 Propiedades de las relaciones

Las relaciones poseen ciertas propiedades que se desprenden directamente de su propia definición:

- No existen tuplas duplicadas. Debido a que el cuerpo de la relación es un conjunto matemático (de tuplas), estos no admiten que existan valores duplicados. Esta primera propiedad sirve para ilustrar el punto de que, en general, una relación y una tabla no son lo mismo, ya que una tabla en sí misma puede contener filas duplicadas, mientras que una relación no admite bajo ningún concepto tuplas duplicadas.
- Las tuplas están en desorden, de arriba hacia abajo. Al igual que la propiedad anterior, ésta surge del hecho que el cuerpo de la relación es

un conjunto matemático: en matemáticas, los conjuntos no están ordenados. Como se mencionó en la propiedad anterior, esta segunda propiedad sirve también para ilustrar la idea de que una relación y una tabla no son lo mismo, ya que las filas de una tabla obviamente tienen un ordenamiento de arriba hacia abajo, mientras que las tuplas de una relación no lo tienen.

- Los atributos están en desorden, de izquierda a derecha. Esta propiedad surge del hecho de que el encabezado de una relación también es un conjunto (de atributos). Por esta razón no existe algo como el "primer atributo", etc. Por lo tanto, siempre se hace referencia a los atributos por nombre y nunca por posición. Esto tiene como resultado la reducción de errores y una programación más clara. Nuevamente esta propiedad marca diferencias entre una relación y una tabla, por la sencilla razón que en una tabla las columnas se encuentran ordenadas, y se pueden acceder a ellas a través de posiciones.

1.2.2.8 Reglas de integridad

Una vez definida la estructura de datos del modelo relacional, es necesario mencionar las reglas de integridad que los datos almacenados en dicha estructura deben cumplir para garantizar que son correctos. Las reglas de integridad son restricciones que definen los estados de consistencia de la Base de Datos.

Al definir cada atributo sobre un dominio se impone una restricción sobre el conjunto de valores permitidos para cada atributo. A este tipo de restricciones se les denomina restricciones de dominios. Hay además dos reglas de integridad muy importantes que son restricciones que se deben cumplir en todas las Bases de Datos relacionales y en todos sus estados o instancias. Antes de definirlas, es preciso conocer el concepto de nulo. Cuando en una tupla un atributo es desconocido, se dice que es nulo. Un nulo no representa el valor cero ni la cadena vacía, éstos son valores que tienen significado. El nulo implica ausencia de información, bien porque al insertar la tupla se desconocía el valor del atributo, o bien porque para dicha tupla el atributo no tiene sentido. Existen dos reglas generales que aporta el modelo relacional:

1. Regla de integridad de las entidades. Ninguno de los atributos que componen la llave primaria puede ser nulo.
2. Regla de integridad referencial. Si en una relación hay llaves foráneas, sus valores deben coincidir con valores de la llave primaria a la que hace referencia.

1.2.2.9 Normalización

La normalización es el proceso mediante el cual se transforman datos complejos a un conjunto de estructuras de datos más pequeños, que además de ser más simples y más estables, son más fáciles de mantener. También se puede entender

la normalización como una serie de reglas que sirven para ayudar a los diseñadores de Bases de Datos a desarrollar un esquema que minimice los problemas de lógica. Cada regla está basada en la que le antecede. La normalización se adoptó porque el viejo estilo de poner todos los datos en un solo lugar, como un archivo o una tabla de la Base de Datos, era ineficiente y conducía a errores de lógica cuando se trataban de manipular los datos.

Otra ventaja de la normalización de Base de Datos es el consumo de espacio. Una Base de Datos normalizada ocupa menos espacio en disco que una no normalizada. Hay menos repetición de datos, lo que tiene como consecuencia un menor uso de espacio en disco.

Existen básicamente tres niveles de normalización: Primera Forma Normal (1NF), Segunda Forma Normal (2NF) y Tercera Forma Normal (3NF). Cada una de estas formas tiene sus propias reglas. Cuando una Base de Datos se conforma a un nivel, se considera normalizada a esa forma de normalización. No siempre es una buena idea tener una Base de Datos conformada en el nivel más alto de normalización, puede llevar a un nivel de complejidad que pudiera ser evitado si estuviera en un nivel más bajo de normalización.

❒ Primera Forma Normal (1FN)

La regla de la Primera Forma Normal establece que las columnas repetidas deben eliminarse y colocarse en tablas separadas.

Un dato sin normalizar no cumple con ninguna regla de normalización. Para explicar con un ejemplo en qué consiste cada una de las reglas, se considera los datos de la tabla 1-1.

ORDEN	FECHA	CLIENTE_ID	NOMBRE_CLIENTE	ESTADO	PRODUCTO_ID	PRODUCTO	CANT	PRECIO
2301	02/02/2006	101	MARTINEZ	CHIAPAS	25	RED	3	35
2301	02/02/2006	101	MARTINEZ	CHIAPAS	10	RAQUETA	6	65
2301	02/02/2006	101	MARTINEZ	CHIAPAS	50	PELOTA	8	4.75
2302	02/03/2006	107	HERNANDEZ	YUCATAN	25	RED	4	5
2303	02/04/2006	110	ESPINOZA	MORELOS	10	RAQUETA	2	65
2303	02/04/2006	110	ESPINOZA	MORELOS	17	FUNDA	2	10

Tabla 0-1

Estos datos contienen un grupo repetido para PRODUCTO, CANT y PRECIO. La 1FN prohíbe los grupos repetidos, por lo tanto se debe convertir a la Primera Forma Normal. Los pasos a seguir son:

- Eliminar los grupos repetidos.
- Crear una nueva tabla con la llave primaria de la tabla base y el grupo repetido.

Los registros quedan ahora conformados en dos tablas: ORDENES (Tabla 1-2) y ARTICULOS_ORDENES (Tabla 1-3)

ORDENES				
ORDEN_ID	**FECHA**	**CLIENTE_ID**	**NOM_CLIENTE**	**ESTADO**
2301	02/02/2006	101	MARTINEZ	CHIAPAS
2302	02/03/2006	107	HERNANDEZ	YUCATAN
2303	02/04/2006	110	ESPINOZA	MORELOS

Tabla 1-2

ARTICULOS_ORDENES				
ORDEN_ID	**PRODUCTO_ID**	**PRODUCTO**	**CANT**	**PRECIO**
2301	25	RED	3	35
2301	10	RAQUETA	6	65
2301	50	PELOTA	8	4.75
2302	25	RED	4	5
2303	10	RAQUETA	2	65
2303	17	FUNDA	2	10

Tabla 1-3

❒ Segunda Forma Normal (2FN)

La regla de la Segunda Forma Normal establece que todas las dependencias parciales se deben eliminar y separar dentro de sus propias tablas. Una dependencia parcial es un término que describe a aquellos datos que no dependen de la llave primaria de la tabla para identificarlos.

Una vez alcanzado el nivel de la Segunda Forma Normal, se controlan la mayoría de los problemas de lógica. Se puede insertar un registro sin un exceso de datos en la mayoría de las tablas.

Siguiendo con el ejemplo, se procede a aplicar la segunda formal normal, es decir, tenemos que eliminar cualquier columna no llave que no dependa de la llave primaria de la tabla. Los pasos a seguir son:

- Determinar cuáles columnas que no son llave no dependen de la llave primaria de la tabla.
- Eliminar esas columnas de la tabla base.
- Crear una segunda tabla con esas columnas y la(s) columna(s) de la llave primaria de la cual dependen.

La tabla ORDENES está en 2FN. Cualquier valor único de ORDEN_ID determina un sólo valor para cada columna. Por lo tanto, todas las columnas son dependientes de la llave primaria ORDEN_ID.

Por su parte, la tabla ARTICULOS_ORDENES no se encuentra en 2FN ya que las columnas PRECIO y PRODUCTO son dependientes de PRODUCTO_ID, pero no son dependientes de ORDEN_ID. Lo que haremos a continuación es eliminar éstas columnas de la tabla ARTICULOS_ORDENES (Tabla 1-4) y crear una tabla ARTICULOS (Tabla 1-5) con dichas columnas y la llave primaria de la que dependen.

Las tablas quedan de la siguiente forma:

ARTICULOS_ORDENES		
ORDEN_ID	**PRODUCTO_ID**	**CANT**
2301	25	3
2301	10	6
2301	50	8
2302	25	4
2303	10	2
2303	17	2

Tabla 0-4

ARTICULOS		
PRODUCTO_ID	**PRODUCTO**	**PRECIO**
25	RED	35
10	RAQUETA	65
50	PELOTA	4.75
3141	FUNDA	10

Tabla 0-5

❒ Tercera Forma Normal (3FN)

Una tabla está normalizada en esta forma si todas las columnas que no son llave son funcionalmente dependientes por completo de la llave primaria y no hay dependencias transitivas. Anteriormente se menciona que una dependencia transitiva es aquella en la cual existen columnas que no son llave que dependen de otras columnas que tampoco son llave.

Cuando las tablas están en la Tercera Forma Normal se previenen errores de lógica cuando se insertan o borran registros. Cada columna en una tabla está identificada de manera única por la llave primaria, y no debe haber datos repetidos. Esto provee un esquema que es fácil de trabajar y expandir.

Para terminar con el ejemplo, la Tercera Forma Normal dice que se debe eliminar cualquier columna no llave que sea dependiente de otra columna no llave. Los pasos a seguir son:

- Determinar las columnas que son dependientes de otra columna no llave.

- Eliminar esas columnas de la tabla base.
- Crear una segunda tabla con esas columnas y con la columna no llave de la cual son dependientes.

Al observar las tablas que se crearon, se observa que la tabla ARTICULOS, como la tabla ARTICULOS_ORDENES se encuentran en 3FN. Sin embargo la tabla ORDENES no lo está, ya que NOM_CLIENTE y ESTADO son dependientes de CLIENTE_ID, y esta columna no es la llave primaria.

Para normalizar esta tabla, se mueven las columnas no llave y la columna llave de la cual dependen dentro de una nueva tabla CLIENTES . Las nuevas tablas ORDENES y CLIENTES se muestran a continuación en las tablas 1-6 y 1-7.

ORDENES		
ORDEN	**FECHA**	**CLIENTE_ID**
2301	02/02/2006	101
2302	02/03/2006	107
2303	02/04/2006	110

Tabla 0-6

CLIENTES		
CLIENTE_ID	**NOM_CLIENTE**	**ESTADO**
101	MARTINEZ	CHIAPAS
107	HERNANDEZ	YUCATAN
110	ESPINOZA	MORELOS

Tabla 0-7

1.3 Sistema Manejador de Bases de Datos

Los sistemas de Bases de Datos separan la definición de la estructura de los datos, de los programas de aplicación y almacenan esta definición en la Base de Datos. Si se añaden nuevas estructuras de datos o se modifican las ya existentes, los programas de aplicación no se ven afectados ya que no dependen directamente de aquello que se ha modificado.

El sistema manejador de la Base de Datos o Data Base Manager System (DBMS) es una aplicación que permite a los usuarios definir, crear y mantener la Base de Datos, y proporciona acceso controlado a la misma.

El DBMS es la aplicación que interacciona con los usuarios de los programas de aplicación y la Base de Datos. En general, un DBMS proporciona los siguientes servicios:

- Permite la definición de la Base de Datos mediante el lenguaje de definición de datos. Este lenguaje permite especificar la estructura y el

tipo de los datos, así como las restricciones sobre los datos. Todo esto se almacenará en la Base de Datos.

- Permite la inserción, actualización, eliminación y consulta de datos mediante el lenguaje de manejo de datos. El hecho de disponer de un lenguaje para realizar consultas reduce el problema de los sistemas de ficheros, en los que el usuario tiene que trabajar con un conjunto fijo de consultas, o bien, dispone de un gran número de programas de aplicación costosos de gestionar.
- Proporciona un acceso controlado a la Base de Datos mediante:
  - Un sistema de seguridad, de modo que los usuarios no autorizados no puedan acceder a la Base de Datos.
  - Un sistema de integridad que mantiene la integridad y la consistencia de los datos.
  - Un sistema de control de concurrencia que permite el acceso compartido a la Base de Datos.
  - Un sistema de control de recuperación que restablece la Base de Datos después de que se produzca un fallo del hardware o del software.
  - Un diccionario de datos o catálogo accesible por el usuario que contiene la descripción de los datos de la Base de Datos.
- El DBMS gestiona la estructura física de los datos y su almacenamiento. Con esta funcionalidad, el DBMS se convierte en una herramienta de gran utilidad. Sin embargo, desde el punto de vista del usuario, se podría discutir que los DBMS han hecho las cosas más complicadas, ya que ahora los usuarios ven más datos de los que realmente quieren o necesitan, puesto que ven la Base de Datos completa. Conscientes de este problema, los DBMS proporcionan un mecanismo de vistas que permite que cada usuario tenga su propia vista o visión de la Base de Datos.

Los DBMS están en continua evolución, tratando de satisfacer los requerimientos de todo tipo de usuarios. Por ejemplo, muchas aplicaciones de hoy en día necesitan almacenar imágenes, vídeo, sonido, etc.

1.3.1 Funciones de los sistemas manejadores de Bases de Datos

Un eficiente DBMS debe cumplir las siguientes funciones:

- La capacidad de almacenar datos en la Base de Datos, acceder a ellos y actualizarlos.
- Debe ocultar al usuario la estructura física interna.
- Proporcionar un catálogo en el que se almacenen las descripciones de los datos y que sean accesibles para los usuarios.
- Debe tener mecanismos que garantice que todas las actualizaciones correspondientes a una determinada transacción se realicen, o que no se realice ninguna. Si la transacción falla durante su realización, por ejemplo porque falla el hardware, la Base de Datos quedará en un estado inconsistente. Algunos de los cambios se habrán hecho y otros no, por lo tanto, los cambios realizados deberán ser deshechos para devolver la Base de Datos a un estado consistente.
- Cuando varios usuarios están usando concurrentemente la Base, el DBMS debe asegurar que está se actualicé correctamente.
- Un DBMS debe proporcionar un mecanismo capaz de recuperar la Base de Datos en caso de que ocurra algún suceso que la dañe. Esto puede ser a causa de un fallo en algún dispositivo hardware o un error del software, que hagan que el DBMS aborte, o puede ser a causa de que el usuario detecte un error durante la transacción y la aborte antes de que finalice. En todos estos casos, el DBMS debe proporcionar un mecanismo capaz de recuperar la Base de Datos llevándola a un estado consistente.
- Es necesario un mecanismo que garantice que sólo los usuarios autorizados pueden acceder a la Base de Datos. La protección debe ser contra accesos no autorizados, tanto intencionados como accidentales.
- Debe ser capaz de integrarse con algún software de comunicación. Muchos usuarios acceden a la Base de Datos desde terminales. En ocasiones estos terminales se encuentran conectados directamente a la máquina sobre la que funciona el DBMS. En otras ocasiones los terminales están en lugares remotos, por lo que la comunicación con la máquina que alberga al DBMS se debe hacer a través de una red. En cualquiera de los dos casos, el DBMS recibe peticiones en forma de mensajes y responde de modo similar.
- La integridad de la Base de Datos requiere la validez y consistencia de los datos almacenados. Un DBMS debe proporcionar los medios necesarios para garantizar que tanto los datos de la Base de Datos, como los cambios que se realizan sobre estos datos, sigan ciertas reglas.

1.4 Arquitectura de los sistemas de Bases de Datos

Hay tres características importantes de los sistemas de Bases de Datos: la separación entre los programas de aplicación y los datos, el manejo de múltiples vistas por parte de los usuarios y el uso de un catálogo para almacenar el esquema de la Base de Datos.

El objetivo de la arquitectura de tres niveles es el de separar los programas de aplicación de la Base de Datos física. En esta arquitectura, mostrada en la figura 1-14, el esquema de una Base de Datos se define en tres niveles de abstracción distintos:

1. En el nivel interno se describe la estructura física de la Base de Datos mediante un esquema interno. Este esquema se especifica mediante un modelo físico y describe todos los detalles para el almacenamiento de la Base de Datos, así como los métodos de acceso.
2. En el nivel conceptual, que s el nivel medio de abstracción, se trata de la representación de los datos realizada por la organización, que recoge las vistas parciales de los requerimientos de los diferentes usuarios y las aplicaciones posibles. Se configura como visión organizativa total, e incluye la definición de datos y las relaciones entre ellos.
3. En el nivel externo se describen varios esquemas externos o vistas de usuario. Cada esquema externo describe la parte de la Base de Datos que interesa a un grupo de usuarios determinados y ocultos a ese grupo el resto de la Base de Datos. En este nivel se puede utilizar un modelo conceptual o un modelo lógico para especificar los esquemas.

Los tres esquemas no son más que descripciones de los mismos datos pero con distintos niveles de abstracción. Los únicos datos que existen realmente están a nivel físico, almacenados en algún dispositivo.

La arquitectura de tres niveles es útil para explicar el concepto de independencia de datos que podemos definir como la capacidad para modificar el esquema en un nivel del sistema sin tener que modificar el esquema del nivel inmediato superior.

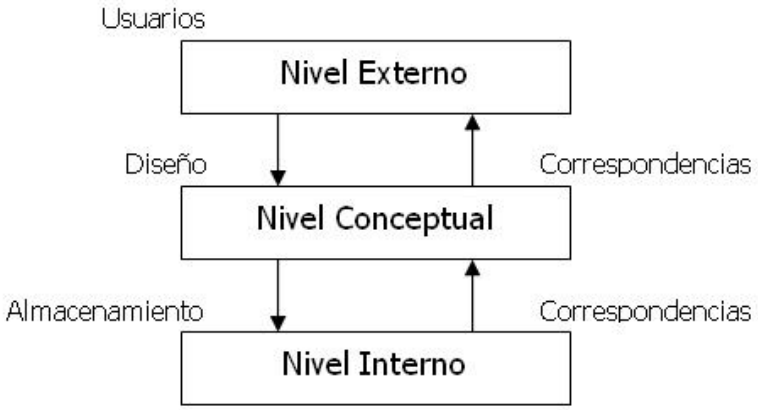


Figura 0-14 Niveles de Arquitectura de Base de Datos

Se pueden definir dos tipos de independencia de datos:

1. La independencia lógica es la capacidad de modificar el esquema conceptual sin tener que alterar los esquemas externos ni los programas de aplicación. Se puede modificar el esquema conceptual para ampliar la Base de Datos o para reducirla. Si, por ejemplo, se reduce la Base de Datos eliminando una entidad, los esquemas externos que no se refieran a ella no deberán verse afectados.
2. La independencia física es la capacidad de modificar el esquema interno sin tener que alterar el esquema conceptual. Dado que la independencia física se refiere sólo a la separación entre las aplicaciones y las estructuras físicas de almacenamiento, es más fácil de conseguir que la independencia lógica.

1.5 Ciclo de vida de las aplicaciones de Bases de Datos

El desarrollo de una Base de Datos se integra dentro de un proceso más amplio que alcanza desde que se toma la iniciativa de crear la Base de Datos hasta que se encuentre totalmente operativa.

La creación de una Base de Datos es, generalmente, una operación difícil, larga y costosa que no puede improvisarse. No se trata solamente de un problema técnico, ya que las repercusiones que esta decisión puede tener en todos los niveles de la empresa (transferencia de responsabilidades entre personas y servicios, reorganización del departamento de Informática, formación de los usuarios, cambios de determinados métodos de trabajo, etc.) hacen de ella una decisión que atañe a la política empresarial, por lo que no debe ser abordada en exclusiva por los técnicos.

La responsabilidad de las decisiones relativas a todo el proceso de creación de una Base de Datos no corresponde únicamente a los informáticos, sino que en ciertas fases, son los directivos y los usuarios.

Las etapas del ciclo de vida de una aplicación de Bases de Datos son las siguientes:

1. Planificación

En esta etapa se analiza el ámbito del proyecto, se hace un estudio de viabilidad, análisis de riesgos, los recursos con los que cuenta una empresa y su presupuesto económico, una planificación temporal y la asignación de los recursos que se necesitaran para todo el proyecto. También deben establecerse estándares de cómo serán los documentos que especifiquen como realizar la recolección de datos. Se debe documentar todos los aspectos legales sobre los datos y los establecidos por la empresa.

2. Análisis

En esta etapa se recogen y analizan los requerimientos de los usuarios y de las áreas de aplicación. Esta información se puede recoger de varias formas:

- Entrevistando al personal de la empresa, concretamente, a aquellos que son considerados expertos en las áreas de interés.
- Observando el funcionamiento de la empresa.
- Examinando documentos, sobre todo aquellos que se utilizan para recoger o visualizar información.
- Utilizando cuestionarios para recoger información de grandes grupos de usuarios.
- Utilizando la experiencia adquirida en el diseño de sistemas similares.

Esta etapa tiene como resultado un conjunto de documentos con las especificaciones de requisitos de los usuarios, en donde se describen las operaciones que se realizan en la empresa desde distintos puntos de vista.

3. Diseño de la Base de Datos

Esta etapa consta de tres fases:

❒ Diseño conceptual

En esta etapa se debe construir un esquema conceptual de la información que se usa en la empresa, independientemente de cualquier consideración física. Esto se representa con un modelo Entidad-Relación.

El primer paso es definir los objetos principales que son de interés al usuario, es decir, las entidades. Estas pueden ser reales o abstractas.

Para iniciar la identificación de entidades se deben tomar en cuenta los siguientes puntos:

- Identificar las actividades que hacen los usuarios
- Identificar cuáles son los datos necesarios que se deben guardar

Se deben documentar los siguientes datos de la entidad en el Esquema Conceptual:

- Nombre de la entidad
- Descripción de la entidad
- Resumen de la entidad
- Observaciones de la entidad

Se deben documentar los siguientes datos del atributo en el Esquema Conceptual:

- Nombre de la entidad
- Nombre del atributo
- Descripción del atributo
- Resumen del atributo
- Tipo de atributo
- Tamaño del atributo
- Observaciones del atributo

Para el proceso de identificación de llaves primarias, primero se deben elegir todos aquellos atributos o conjuntos mínimos de atributos que identifiquen de manera específica a cada registro (ocurrencia), es decir, elegir las llaves candidatas y determinar si es la principal forma de acceso a los datos que almacena. Se deben identificar y documentar todas las relaciones existentes entre las distintas entidades. Por lo general, un verbo o preposición que asocie entidades se traduce en una relación. Se puede hablar de tres tipos de relación:

- Relaciones de existencia (Ej. Jefe tiene Empleados)
- Relaciones funcionales (Ej. Jefe reporta Incidencias)
- Relaciones de eventos (Ej. Jefe paga Nomina)

Además de la anterior selección y clasificación, se debe identificar cuál entidad es la entidad padre y cuál es la entidad hija. Se debe definir la existencia de las entidades hijas y también su cardinalidad. Además, se debe especificar si la entidad hija hereda la llave primaria de la entidad padre.

Una vez finalizada la construcción de un modelo de datos para un área funcional se debe integrar con los otros modelos que se vayan

construyendo para ir conformando así el modelo de datos del área de estudio en su totalidad.

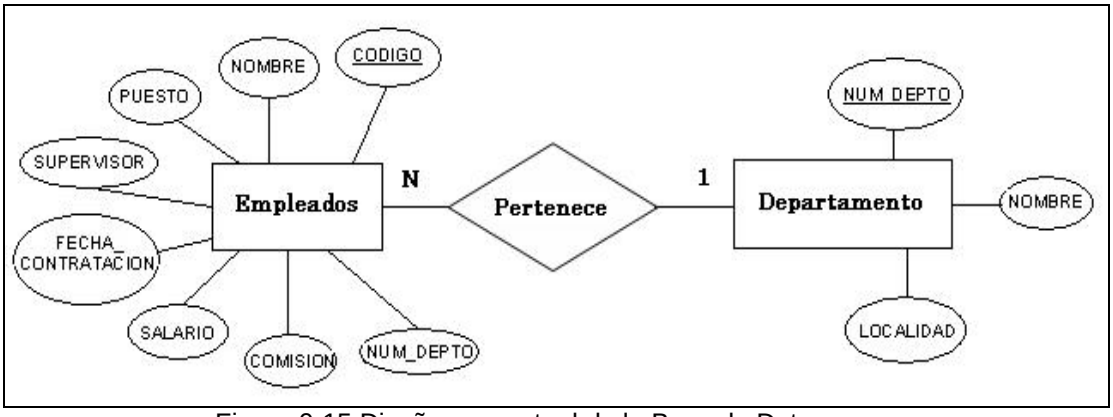


Figura 0-15 Diseño conceptual de la Base de Datos

❒ Diseño lógico

El diseño lógico es el proceso de construir un esquema lógico de la información que utiliza la empresa, basándose en un modelo de Base de Datos específico, independiente del DBMS concreto que se vaya a utilizar y de cualquier otra consideración física.

En esta etapa, se transforma el esquema conceptual en un esquema lógico que utilizará las estructuras de datos del modelo de Base de Datos en el que se basa el DBMS que se vaya a utilizar. Conforme se va desarrollando el esquema lógico, éste se va probando y validando con los requisitos de usuario.

Básicamente consiste en trasladar las entidades de los diagramas Entidad-Relación a tablas relacionales para llevar a cabo el proceso de normalización.

El diagrama que se obtuvo en el diseño conceptual se transforma al siguiente:

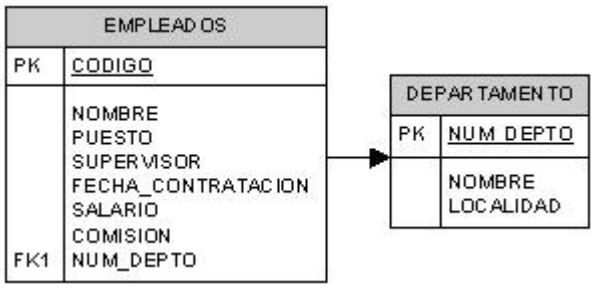


Figura 0-16 Diseño lógico de la Base de Datos.

- ❒ Diseño físico

  Esta etapa, se parte del esquema lógico global obtenido durante el diseño lógico y se obtiene una descripción de la implementación de la Base de Datos. Esta descripción es completamente dependiente del DBMS específico que se vaya a utilizar.

4. Diseño de la aplicación

En esta etapa se diseñan los programas de aplicación que usarán y procesarán la Base de Datos. En la mayor parte de los casos no se puede finalizar el diseño de las aplicaciones hasta que se ha terminado con el diseño de la Base de Datos. Por otro lado, la Base de Datos existe para dar soporte a las aplicaciones, por lo que habrá una realimentación desde el diseño de las aplicaciones al diseño de la Base de Datos.

Hay que asegurarse de que toda la funcionalidad especificada en los requisitos de usuario se encuentra en el diseño de la aplicación. Habrá algunos programas que utilicen y procesen los datos de la Base de Datos.

Además, habrá que diseñar las interfaces de usuario para que sea fácil de operar.

5. Implementación

En esta fase final se hace realidad la Base de Datos, mediante la creación y la compilación del esquema de Bases de Datos y de los ficheros de Bases de Datos, así como de las transacciones, a través de las aplicaciones. Se crean las tablas y las reglas de tal manera que se asegure la integridad referencial. También la creación de consultas que permitan la visualización de una información determinada y la elaboración de informes que preparen la información requerida para su presentación a través de impresora.

6. Prueba

En esta etapa se prueba y valida el sistema con los requisitos especificados por los usuarios. Para ello, se debe diseñar un set de datos reales, que se deben llevar a cabo de manera metódica y rigurosa. Es importante darse cuenta de que la fase de prueba no sirve para demostrar que no hay fallos, sirve para encontrarlos. Si la fase de prueba se lleva a cabo correctamente, descubrirá los errores en los programas de aplicación y en la estructura de la Base de Datos.

7. Mantenimiento

Una vez que el sistema está completamente implementado y probado, se pone a disposición para ser usado. El sistema está ahora en la fase de mantenimiento en la que se llevan a cabo las siguientes tareas:

- Monitoreo de eficiencia del sistema. Si la eficiencia cae por debajo de un determinado nivel establecido, puede ser necesario reorganizar la Base de Datos.
- Mantenimiento y actualización del sistema. Cuando sea necesario, los nuevos requisitos que vayan surgiendo se incorporarán al sistema, siguiendo de nuevo las etapas del ciclo de vida que se acaban de presentar.

1.6 Lenguaje del sistema manejador de Bases de Datos

Los DBMS deben ofrecer lenguajes e interfaces apropiadas para cada tipo de usuario: administradores de la Base de Datos, diseñadores, programadores de aplicaciones y usuarios finales.

1.6.1 Lenguaje de Consulta Estructurado

Los sistemas comunes de Base de Datos son relacionales consultan y modifican la Base por medio de un lenguaje denominado Structured Query Language (Lenguaje de Consulta Estructurado o SQL).

El SQL es tanto un lenguaje de definición de datos (Data Definition Language o DDL) como un lenguaje de manipulación de datos (Data Manipulation Language o DML). Además, cuenta con mecanismos para definir vistas de la Base de Datos, para especificar seguridad y autorización, para definir restricciones de integridad, y para especificar controles de transacciones. También tiene reglas para insertar sentencias de SQL en lenguajes de programación de propósito general como C, Pascal o Java.

El lenguaje SQL está compuesto por comandos, cláusulas, operadores y funciones de agregado. Estos elementos se combinan en las instrucciones para crear, actualizar y manipular las Bases de Datos.

Los tipos de comandos SQL se describen a continuación:

1.6.1.1 Lenguaje de Definición de Datos.

Una vez finalizado el diseño de una Base de Datos y escogido un DBMS para su implementación, el primer paso consiste en especificar el esquema conceptual y el esquema interno de la Base de Datos, y la correspondencia entre ambos. En muchos DBMS no se mantiene una separación estricta de niveles, por lo que el administrador de la Base de Datos y los diseñadores utilizan el mismo lenguaje para definir ambos esquemas: el lenguaje de definición de datos (DDL).

El DBMS posee un compilador de DDL cuya función consiste en procesar las sentencias del lenguaje para identificar las descripciones de los distintos

elementos de los esquemas y almacenar la descripción del esquema en el catálogo o diccionario de datos.

- **Creación de tablas**

La sintaxis básica de creación de una tabla es la siguiente:

```
CREATE TABLE [esquema.]tabla
([atrib1 tipo1 [DEFAULT definicion]] [NOT NULL]
[,atrib2 tipo2 [DEFAULT definicion] [NOT NULL]]...
[CONSTRAINT nombre_restr][UNIQUE|PRIMARY KEY (atrib1[,atrib2]...)]
[CONSTRAINT nombre_restr][FOREIGN KEY (atrib1 [,atrib2]...)
REFERENCES [esquema.]tabla(atrib1[,atrib2]...)]
[CONSTRAINT nombre_restr] [CHECK condicion]
[[ON DELETE | ON UPDATE] [SET NULL|SET DEFAULT|CASCADE]]
[AS consulta])
```

Ejemplo 1

```
CREATE TABLE departamento
(num_depto NUMBER(4),
nombre VARCHAR(20),
localidad VARCHAR(20))
CONSTRAINT pk_num_depto PRIMARY KEY (num_depto)

CREATE TABLE empleados
(
codigo NUMBER(4),
nombre VARCHAR(50),
puesto VARCHAR(25),
supervisor NUMBER(2),
fecha_contratacion DATE,
salario NUMBER(6),
comision NUMBER(4),
num_depto NUMBER(2)
)
CONSTRAINT pk_codigo PRIMARY KEY (codigo)
CONSTRAINT fk_num_depto FOREIGN KEY (num_depto)
   REFERENCES DEPARTAMENTO
 (num_depto))
```

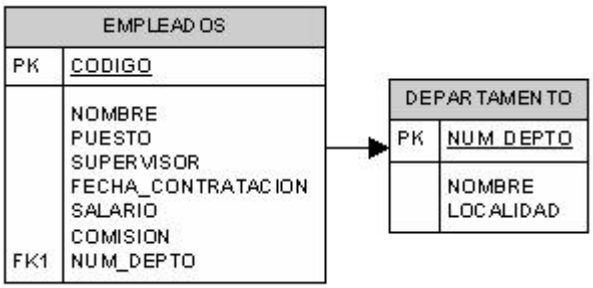


Figura 0-17 Ejemplo: Creación de tablas

1.6.1.2 Lenguaje de Manejo de Datos

Permiten generar consultas para ordenar, filtrar y extraer datos de la Base de Datos. Una vez creados los esquemas de la Base de Datos, los usuarios necesitan un lenguaje que les permita manipular los datos de la Base de Datos: realizar consultas, inserciones, eliminaciones y modificaciones.

- **Consulta de datos**

La sintaxis básica de una consulta de selección es la siguiente:

SELECT Campos FROM Tabla;

Ejemplo 2

La tabla EMPLEADOS contiene las columnas:

Codigo= Código de empleado
Nombre= Nombre de empleado
Puesto= Puesto de trabajo
Supervisor= Código de supervisor
Fecha_Contratacion= Fecha de contratación
Salario= Salario
Comision= Comision
Num_Depto= Número de departamento donde trabaja el empleado

Respecto a la tabla DEPARTAMENTO se tiene:

Num_depto= Número de departamento
Nombre= Nombre del departamento
Localidad= Localidad donde se encuentra el departamento

La consulta más sencilla es seleccionar todas las tuplas de una tabla:
"Seleccionar los datos de todos los empleados", como se muestra en la tabla 1-8.

SELECT codigo, nombre, puesto,supervisor, fecha_contratacion, salario,comision, num_depto
FROM empleados

Se utiliza el asterisco (*) como comodín para seleccionar todos los campos. La sentencia anterior se observa en la tabla 1-9 y es equivalente a:

SELECT num_depto, nombre, localidad
FROM departamento

CODIGO	NOMBRE	PUESTO	SUPERVISOR	FECHA_CONTRATACION	SALARIO	COMISION	NUM_DEPTO
3	ALFONSO SEPULVEDA	VENDEDOR	21	12/02/1990	4000	1000	2
4	JUAN MANUEL ALONSO	ADMINISTRATIVO	23	06/02/2000	5000		3
5	IRMA LETICIA GUTIERREZ	ANALISTA	20	20/07/1998	7500		1
6	JUANITA MEDINA	ANALISTA	20	30/11/1997	7500		1
7	JOSE LUIS SILVA	ADMINISTRATIVO	23	09/03/2005	5000		3
8	JUANA GONZALEZ	VENDEDOR	21	23/09/1987	4000	1000	2
9	EMILIA VALADEZ	ANALISTA	20	07/02/1999	7500		1
10	JUAN ANTONIO MANRIQUEZ	VENDEDOR	21	29/10/2003	4000	1000	2
11	AURELIA ESPARZA	ANALISTA	20	04/01/1994	7500		1
12	BENITO MACIAS	VENDEDOR	21	08/06/1996	4000	1000	2
13	RAUL REVELES	ADMINISTRATIVO	23	09/03/2005	5000		3
14	JORGE LARA	VENDEDOR	21	19/06/1996	4000	1000	2
15	PETRA RAMOS	ANALISTA	20	22/08/1992	7500		1
16	OSCAR DIAZ	VENDEDOR	21	20/11/2001	4000	1000	2
17	CAROLINA HERNANDEZ	VENDEDOR	21	03/12/2002	4000	1000	2
18	MA DE LOURDES ARCE	VENDEDOR	21	30/11/1993	4000	1000	2
19	ROBERTO GONZALEZ	VENDEDOR	21	01/09/1989	4000	1000	2
20	MIGUEL ANGEL CARDONA	ADMINISTRATIVO	23	08/07/2004	5000		3

Tabla 0-8

NUM_DEPTO	NOMBRE	LOCALIDAD
1	OPERACION	DF
2	VENTAS	GUANAJUATO
3	ADMINISTRACION	GUANAJUATO

Tabla 0-9

Para seleccionar una o varias columnas véase la siguiente sentencia (Ejemplo en la tabla 1-10):

SELECT codigo, nombre
FROM empleados

CODIGO	NOMBRE
1	MA GUADALUPE RODRIGUEZ
2	ROBERTO RAMIREZ
3	ALFONSO SEPULVEDA
4	JUAN MANUEL ALONSO
5	IRMA LETICIA GUTIERREZ
6	JUANITA MEDINA
7	JOSE LUIS SILVA
8	JUANA GONZALEZ
9	EMILIA VALADEZ
10	JUAN ANTONIO MANRIQUEZ
11	AURELIA ESPARZA
12	BENITO MACIAS

Tabla 0-10

Cláusula WHERE

Para restringir las filas que se obtienen, se pueden imponer condiciones. Para ello se usa la cláusula WHERE. Cuando SQL aplica una cláusula WHERE a una tabla, se omiten todas las filas de dicha tabla para las cuales la expresión de WHERE es falsa o nula. En la tabla 1-11 se pueden observar los diferentes operadores usados para esta cláusula.

Operador	Significado
<	Menor que
>	Mayor que
<>	Distinto de
<=	Menor ó Igual que
>=	Mayor ó Igual que
=	Igual que
Between	Especifica un rango de valores.
Like	Comparación de un modelo
In	Especificar varios registros de una Base de Datos

Tabla 0-11

Como ejemplo, para listar los datos de los empleados cuyo salario sea mayor o igual a 5000 se utilizaría la siguiente sentencia que da como resultado lo mostrado en la tabla 1-12.

```
SELECT nombre,puesto,salario,num_depto
  FROM empleados
WHERE salario>=5000
```

NOMBRE	PUESTO	SALARIO	NUM_DEPTO
MA GUADALUPE RODRIGUEZ	ANALISTA	7500	1
JUAN MANUEL ALONSO	ADMINISTRATIVO	5000	3
IRMA LETICIA GUTIERREZ	ANALISTA	7500	1
JUANITA MEDINA	ANALISTA	7500	1
JOSE LUIS SILVA	ADMINISTRATIVO	5000	3
EMILIA VALADEZ	ANALISTA	7500	1
AURELIA ESPARZA	ANALISTA	7500	1
RAUL REVELES	ADMINISTRATIVO	5000	3
PETRA RAMOS	ANALISTA	7500	1
MIGUEL ANGEL CARDONA	ADMINISTRATIVO	5000	3

Tabla 0-12

❒ Relaciones entre tablas

La tabla EMPLEADOS posee una llave foránea que es la llave primaria de la tabla DEPARTAMENTO, el cruce se hace por estas columnas. A cada tabla se le asigna un alias para poder distinguir la columna de cada tabla:

Para obtener aquellos trabajadores que laboran en Guanajuato se ejecuta la sentencia:

```
SELECT e.codigo,e.nombre,e.num_depto,d.localidad
   FROM empleados e, departamento d
 WHERE d.num_depto=e.num_depto
    AND localidad='GUANAJUATO'
```

CODIGO	NOMBRE	NUM_DEPTO	LOCALIDAD
2	ROBERTO RAMIREZ	2	GUANAJUATO
3	ALFONSO SEPULVEDA	2	GUANAJUATO
4	JUAN MANUEL ALONSO	3	GUANAJUATO
7	JOSE LUIS SILVA	3	GUANAJUATO
8	JUANA GONZALEZ	2	GUANAJUATO
10	JUAN ANTONIO MANRIQUEZ	2	GUANAJUATO
12	BENITO MACIAS	2	GUANAJUATO
13	RAUL REVELES	3	GUANAJUATO
14	JORGE LARA	2	GUANAJUATO
16	OSCAR DIAZ	2	GUANAJUATO
17	CAROLINA HERNANDEZ	2	GUANAJUATO
18	MA DE LOURDES ARCE	2	GUANAJUATO
19	ROBERTO GONZALEZ	2	GUANAJUATO
20	MIGUEL ANGEL CARDONA	3	GUANAJUATO

Tabla 0-13

❒ Inserción de datos

Para insertar datos en una tabla se utiliza la sentencia INSERT:

En este caso la sintaxis es la siguiente:

```
INSERT INTO Tabla (campo1, campo2, .., campoN)
VALUES (valor1, valor2, ..., valorN)
```

Ejemplo 3

```
INSERT INTO empleados (codigo, nombre, puesto,supervisor,
fecha_contratacion, salario,comision, num_depto)
VALUES ( 21,'RICARDO HERNANDEZ','VENDEDOR',21,'12/08/1996','4000' ,
1000 ,2 )
```

❒ Modificación de datos

Para modificar los datos se utiliza la sentencia UPDATE. Su sintaxis es:

```
UPDATE Tabla
      SET Campo1=Valor1, Campo2=Valor2, ... CampoN=ValorN
 WHERE Criterio;
```

Así, si se requiere modificar la comisión para los vendedores se utilizaría:

```
UPDATE empleados
     SET comision=1500
WHERE puesto='VENDEDOR'
```

❒ Borrado de datos

Para eliminar los registros se utiliza la sentencia DELETE, puede ser de una o más tablas listadas en la cláusula FROM que satisfagan la cláusula WHERE. Esta consulta elimina los registros completos, no es posible eliminar el contenido de algún campo en concreto. Si se quiere vaciar toda la tabla se omite la cláusula WHERE Su sintaxis es:

```
DELETE campo
  FROM tabla
WHERE criterio
```

Como ejemplo, se eliminara un registro:

```
DELETE empleados
WHERE codigo=1
```

1.7 Ventajas e inconvenientes de los sistemas de Bases de Datos

Los sistemas de Bases de Datos presentan numerosas ventajas que se pueden dividir en dos grupos: las que se deben a la integración de datos y las que se deben a la interfaz común que proporciona el DBMS.

1.7.1 Ventajas por la integración de datos

- Control sobre la redundancia de datos. Los sistemas de ficheros almacenan varias copias de los mismos datos en ficheros distintos. Esto hace que se desperdicie espacio de almacenamiento, además de provocar la falta de consistencia de datos. En los sistemas de Bases de Datos todos estos ficheros están integrados, por lo que no se almacenan varias copias de los mismos datos. Sin embargo, en una Base de Datos no se puede eliminar la redundancia completamente, ya que en ocasiones es necesaria para modelar las relaciones entre los datos.
- Consistencia de datos. Eliminando o controlando las redundancias de datos se reduce en gran medida el riesgo de que haya inconsistencias. Si un dato está almacenado una sola vez, cualquier actualización se debe realizar sólo una vez, y está disponible para todos los usuarios inmediatamente. Si un dato está duplicado y el sistema conoce esta redundancia, el propio sistema puede encargarse de garantizar que todas las copias se mantienen consistentes.
- Más información sobre la misma cantidad de datos. Al estar todos los datos integrados, se puede extraer información adicional sobre los mismos.
- Compartir datos. En los sistemas de ficheros, los ficheros pertenecen a las personas o a los departamentos que los utilizan. Pero en los sistemas de Bases de Datos, la Base de Datos pertenece a la empresa y puede ser compartida por todos los usuarios que estén autorizados. Además, las nuevas aplicaciones que se vayan creando pueden utilizar los datos de la Base de Datos existente.
- Mantenimiento de estándares. Gracias a la integración es más fácil respetar los estándares necesarios, tanto los establecidos a nivel de la empresa como los nacionales e internacionales. Estos estándares pueden establecerse sobre el formato de los datos para facilitar su intercambio, pueden ser estándares de documentación, procedimientos de actualización y también reglas de acceso.

1.7.2 Ventajas por la existencia del DBMS

- Mejora en la integridad de datos. La integridad de la Base de Datos se refiere a la validez y la consistencia de los datos almacenados. Normalmente, la integridad se expresa mediante restricciones o reglas que no se pueden violar. Estas restricciones se pueden aplicar tanto a los datos,

como a sus relaciones, y es el DBMS quien se debe encargar de mantenerlas.

- Mejora en la seguridad. Se refiere a la protección de la Base de Datos frente a usuarios no autorizados. Sin medidas de seguridad eficientes, la integración de datos en los sistemas de Bases de Datos hace que éstos sean más vulnerables que en los sistemas de ficheros. Sin embargo, los DBMS permiten mantener la seguridad mediante el establecimiento de claves para identificar al personal autorizado a utilizar la Base de Datos. Las autorizaciones se pueden realizar a nivel de operaciones, de modo que un usuario puede estar autorizado a consultar ciertos datos pero no a actualizarlos, por ejemplo.
- Mejora en la accesibilidad a los datos. Muchos DBMS proporcionan lenguajes de consultas o generadores de informes que permiten al usuario hacer cualquier tipo de consulta sobre los datos, sin que sea necesario que un programador escriba una aplicación que realice tal tarea.
- Mejora en el mantenimiento gracias a la independencia de datos. En los sistemas de ficheros, las descripciones de los datos se encuentran inmersas en los programas de aplicación que los manejan. Esto hace que los programas sean dependientes de los datos, de modo que un cambio en su estructura, o un cambio en el modo en que se almacena en disco, requiere cambios importantes en los programas cuyos datos se ven afectados. Sin embargo, los DBMS separan las descripciones de los datos de las aplicaciones. Esto es lo que se conoce como independencia de datos, gracias a la cual se simplifica el mantenimiento de las aplicaciones que acceden a la Base de Datos.
- Aumento de la concurrencia. En algunos sistemas de ficheros, si hay varios usuarios que pueden acceder simultáneamente a un mismo fichero, es posible que el acceso interfiera entre ellos de modo que se pierda información o, incluso, que se pierda la integridad. La mayoría de los DBMS gestionan el acceso concurrente a la Base de Datos y garantizan que no ocurran problemas de este tipo.
- Mejora en los servicios de copias de seguridad y de recuperación ante fallos. Muchos sistemas de ficheros dejan que sea el usuario quien proporcione las medidas necesarias para proteger los datos ante fallos en el sistema o en las aplicaciones. Los usuarios tienen que hacer copias de seguridad cada día, y si se produce algún fallo, utilizar estas copias para restaurarlos. En este caso, todo el trabajo realizado sobre los datos desde que se hizo la última copia de seguridad se pierde y se tiene que volver a realizar. Sin embargo, los DBMS actuales funcionan de modo que se minimiza la cantidad de trabajo perdido cuando se produce un fallo.

1.7.3 Inconvenientes de los sistemas de Bases de Datos

- Complejidad. Los DBMS son conjuntos de programas muy complejos con una gran funcionalidad. Es preciso comprender muy bien esta funcionalidad para poder sacar un buen partido de ellos.
- Tamaño. Los DBMS son programas complejos y muy extensos que requieren una gran cantidad de espacio en disco y de memoria para trabajar de forma eficiente.
- Costo económico del DBMS. El costo de un DBMS varía dependiendo del entorno y de la funcionalidad que ofrece. Además, puede pagarse una cuota anual de mantenimiento que suele ser un porcentaje del precio del DBMS.
- Costo del equipamiento adicional. Tanto el DBMS, como la propia Base de Datos, pueden hacer que sea necesario adquirir más espacio de almacenamiento. Además, para alcanzar las prestaciones deseadas, es posible que sea necesario adquirir una máquina más grande o una máquina que se dedique solamente al DBMS. Todo esto hará que la implantación de un sistema de Bases de Datos sea más cara.
- Costo de la conversión. En algunas ocasiones, el costo del DBMS y el costo del equipo informático que sea necesario adquirir para su buen funcionamiento, es insignificante comparado al costo de convertir la aplicación actual en un sistema de Bases de Datos. Este costo incluye el costo de enseñar a la plantilla a utilizar estos sistemas y, probablemente, el costo del personal especializado para ayudar a realizar la conversión y poner en marcha el sistema. Este costo es una de las razones principales por las que algunas empresas y organizaciones se resisten a cambiar su sistema actual de ficheros por un sistema de Bases de Datos.
- Vulnerable a los fallos. El hecho de que todo esté centralizado en el DBMS hace que el sistema sea más vulnerable ante los fallos que puedan producirse.

En este apartado se presentaron los conocimientos básicos para entender las Bases de Datos y su utilidad, con esto, en el siguiente capítulo se podrá abordar el tema de la Ingeniería Inversa y con ello, la posibilidad de enriquecer o evolucionar una Base de Datos.

Ingeniería Inversa de Base de Datos

Las necesidades de evolución y adaptación a los nuevos requerimientos tecnológicos y de negocio empujan a las empresas que manejan la información a través de un software a buscar formas alternativas para no quedarse atrás ante la eminente situación de que el mantenimiento clásico a los sistemas no es suficiente. En una primera aproximación, la solución a este problemas se puede presentar a través de dos caminos: un nuevo desarrollo que incorpore nuevas tecnologías y funcionalidades o por medio de la aplicación de la Reingeniería.

En muchas organizaciones hay Bases de Datos que han evolucionado con el paso del tiempo. En estos sistemas, la comprensión de los datos se ha perdido parcial o totalmente o bien ya no refleja el universo de la información actual, restringiendo por esta razón el uso efectivo que la organización puede hacer de su información.

La conversión de un esquema de Bases de Datos a otro con mayor contenido semántico es un tema de investigación actual, y un tópico con aplicación en varias áreas de desarrollo tales como integración de esquemas, migración de sistemas heredados[1] (legacy system) y Reingeniería de modelos de datos que no están actualizados o de pobre calidad.

La Reingeniería es el proceso de aplicar en primer lugar, Ingeniería Inversa a un sistema software existente para obtener estructuras de mayor nivel de abstracción para, en segundo lugar, aplicar una fase de ingeniería directa mediante la cual se obtenga una nueva versión del sistema con nuevas funcionalidades.

Dentro del proceso de Reingeniería, las técnicas de Ingeniería Inversa son utilizadas para la comprensión de la estructura y el comportamiento de un sistema de software del que no se dispone documentación, o que en caso de disponerse se encuentra obsoleta o es insuficiente.

Mediante el proceso de Reingeniería, y dentro de éste, en la Ingeniería Inversa, no se pretende exclusivamente recuperar especificaciones a más alto nivel del sistema heredado. Además de éste tipo de información, también se pretende obtener documentación, información que resulta importante en el momento de analizar éste tipo de sistemas software y mejorar su comprensión y mantenimiento. Ésta información en la mayoría de las ocasiones, resulta ser inexistente o se ha perdido con el paso del tiempo, ya que a veces los ciclos de desarrollo son rápidos y poco planeados y esto hace que se pierda el desarrollo de documentación asociada, lo que hace que su posterior etapa de mantenimiento sea más difícil y costosa.

[1] Un sistema heredado es aquel que se está utilizando y al cuál se pretende aplicar reingeniería.

2.1 Evolución de los sistemas de información

La evolución de un sistema es un concepto amplio, abarca desde una simple modificación para corregir un error de un programa hasta una reimplantación completa del sistema. Las actividades de evolución que se pueden realizar sobre un sistema son de tres tipos: mantenimiento, abandono y Reingeniería.

2.1.1 Mantenimiento

Históricamente, el mantenimiento se puede considerar como la primera forma de evolución de los sistemas.

El mantenimiento consta de las modificaciones de los productos de software después de su entrega para corregir fallos, mejorar rendimiento u otros atributos o adaptar el producto a un cambio de entorno. [2]

Una definición similar es dada por ISO/IEC[3]: Un producto de software soporta una modificación en el código y su documentación asociada para la solución de un problema o por la necesidad de una mejora. Su objetivo es mejorar el software existente manteniendo su integridad.

El mantenimiento representa el porcentaje más alto del costo de todo el ciclo de vida de un sistema. Se proponen cuatro tipos de mantenimiento:

1. Adaptativo. Su objetivo es modificar un programa para adaptarlo a los cambios hardware o software en el entorno en el que se ejecuta. Puede ser, desde un pequeño cambio, hasta una reescritura de todo el código, y es cada vez más habitual debido a la actualización frecuente los sistemas operativos.
2. Perfectivo. Conjunto de actividades para mejorar o añadir nuevas funcionalidades requeridas por el usuario.
3. Correctivo. Es el conjunto de actividades dedicadas a corregir defectos en el hardware o en el software detectados por los usuarios durante la explotación del sistema. Estos defectos se les conoce como fallos y ocurren cuando el comportamiento de un sistema es diferente al establecido en la especificación. Los tipos de fallos son:

    - Procesamiento: salidas incorrectas de un programa.
    - Rendimiento: tiempo de respuesta demasiado alto en una búsqueda de información.
    - Programación: inconsistencias en el diseño de un programa.

[2] Definición de El ANSI/IEE (American National Standards Institute o El Instituto Nacional Estadounidense de Estándares/ Institute of Electrical and Electronics Engineers o Instituto de Ingenieros Eléctricos y Electrónicos)
[3] International Organization for Standardization o Organización Internacional para la Estandarización /International Electrotechnical Comission o La Comisión Electrotécnica Internacional

- Documentación: inconsistencias entre la documentación de un programa y el manual de usuario del mismo.

4. Preventivo. Conjunto de actividades para facilitar el mantenimiento futuro del sistema.

Si los cambios se pudieran anticipar en el diseño, se podrían prever en forma de obtener parámetros. El problema fundamental es que, el elevado número de cambios que el software soporta a lo largo de su ciclo de vida, hace imposible una previsión en el diseño del sistema. Por tanto, el mantenimiento debe tener una consideración especial porque:

- Consume una gran parte del costo total del ciclo de vida del software.
- Imposibilita un cambio rápido y fiable en el software haciendo perder oportunidades de negocio.
- Dificulta la adopción de nuevas tecnologías.
- Ocasiona daños en la integridad del sistema.

2.1.2 Abandono

Un sistema se abandona cuando no es capaz de seguir el ritmo de las necesidades del negocio y su Reingeniería no es posible o no se puede hacer con un costo efectivo. Normalmente, estos sistemas carecen de documentación, de actualización y de capacidad de expansión.

El abandono de un sistema produce riesgos:

- Es una actividad que consume muchos recursos. Estos recursos estarán ocupados en el mantenimiento del sistema y, seguramente, no conocerán las nuevas tecnologías que van a ser aplicadas en el desarrollo del nuevo sistema.
- Abandonar el sistema no significa empezar un desarrollo desde cero. No se puede pensar que los datos van a desecharse y que los valores que conservan los expertos del sistema van a perderse. El abandono del sistema debe venir precedido por un análisis exhaustivo que permitan decidir el método que se va a utilizar para la migración de la información del sistema actual al nuevo, incluyendo datos, funciones de negocio y arquitectura del sistema.

2.1.3 Reingeniería

Este apartado debe comenzar con la definición de ingeniería. Esta puede definirse como la aplicación de los conocimientos científicos a la invención, perfeccionamiento y utilización de la técnica industrial. Es la función coherente que un grupo de expertos aporta al proceso creador de una realización técnica. Reingeniería entonces es, la revisión de esos procesos, a fin de hacerlos mucho más efectivos.

Reingeniería es la transformación sistemática de un sistema existente a una nueva forma para realizar mejoras de la calidad en operación, capacidad del sistema, funcionalidad, rendimiento o capacidad de evolución a bajo costo, con un plan de desarrollo corto y con bajo riesgo para el cliente[4].

En esta definición se enfatiza que el hecho de que la Reingeniería es la mejora de sistemas existentes de modo que la inversión resulte muy rentable y que, de todas formas, dicha mejora podría ser obtenida a través de un nuevo desarrollo. Para considerar aplicar Reingeniería, esta debe significar: un costo bajo, poco tiempo de desarrollo e implementación, poco riesgo y no existir un valor añadido al cliente. En caso contrario, debe analizarse la posibilidad de un nuevo desarrollo.

La Reingeniería es una alternativa útil para la industria del software, ya que se trata de una actividad que permita incrementar la facilidad de mantenimiento, reutilización y evolución de sistemas de software. En concreto, su aplicación a las Bases de Datos es una necesidad que surge muy a menudo.

Por ello, es previsible que la investigación en el campo del mantenimiento del software vaya en aumento en los próximos años, complementada por la Reingeniería de sistemas.

La Reingeniería trabaja todos los niveles de abstracción, desde la implementación hasta el diseño, para conseguir realizar cambios más drásticos preservando los valores de negocio del sistema.

La Reingeniería de sistemas puede clasificarse según los niveles de conocimientos requeridos para llevar a cabo el proyecto. La Reingeniería que requiere conocimientos a bajos niveles de abstracción (código fuente) se llama Ingeniería Inversa y aquella que sólo requiere el conocimiento de las interfaces del sistema se llama Reingeniería propiamente.

Ante la necesidad de recuperación del diseño, la arquitectura y la funcionalidad del sistema desde el código para modernizar el lenguaje de programación o el soporte de los datos, surge la Ingeniería Inversa. La Ingeniería Inversa es un proceso que sirve para analizar un sistema existente con el objetivo de identificar sus componentes y sus relaciones y crear otras representaciones del mismo o alguna abstracción sobre el propio sistema. Por ejemplo, se puede tomar un programa sin tener el código fuente, ejecutarlo con diversas entradas y analizar sus salidas para crear otro programa similar o idéntico.

[4]La definición dada por el Reengineering Center del Software Engineering Institute de la Universidad Carnegie Mellon

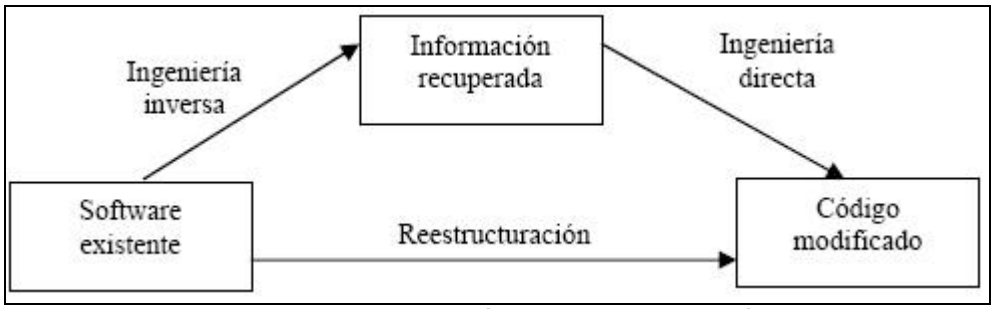


Figura 2-1 Proceso básico de la Reingeniería

2.2 Ingeniería Inversa

El objetivo de la Ingeniería Inversa es obtener información técnica a partir de un producto accesible al público, con el fin de determinar de qué está hecho, qué lo hace funcionar y cómo fue fabricado. Los productos más comunes que son sometidos a la Ingeniería Inversa son los programas de computadoras y los componentes electrónicos.

Este método es denominado Ingeniería Inversa porque avanza en dirección opuesta a las tareas habituales de ingeniería, que consisten en utilizar datos técnicos para elaborar un producto determinado. En general si el producto u otro material que fue sometido a la Ingeniería Inversa fue obtenido en forma apropiada, entonces el proceso es legítimo y legal[5].

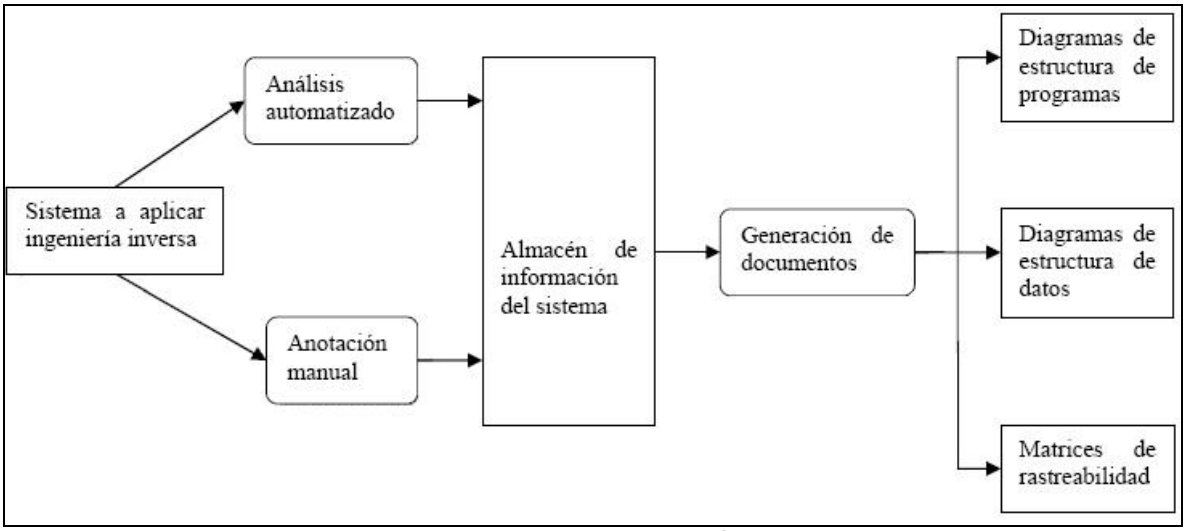


Figura 2-2 Proceso de Ingeniería Inversa

[5] Los procesos legítimos de la ingeniería inversa se analiza en el capítulo

2.2.1 Aplicaciones y limitaciones de la Ingeniería Inversa

Dado que es una labor estratégica, es necesario conocer cuando conviene realizar la tarea de Ingeniería Inversa y Reingeniería y cuando es más rentable sustituirla e implementar un nuevo sistema. Si se presentan las siguientes condiciones, se debe considerar la aplicación de la Ingeniería Inversa:

- El sistema es de una importancia crítica para la empresa que la está utilizando.
- Cuando los recursos utilizados producen un alto costo económico o de personal.
- Si se presentan fallos frecuentes, que muchas veces son difíciles de encontrar o se requiere añadir otras funcionalidades o hay una dificultad para la integración de las mismas, como pueden ser añadir nuevos requisitos, siempre y cuando los requisitos básicos se mantengan.
- Si requiere un mantenimiento frecuente o solo pocas personas están capacitadas para realizar dicho mantenimiento o modificaciones.
- Cuando no existe especificaciones de diseño, no hay cumplimiento de las mismas o simplemente la documentación es inexistente o incomprensible.

Dentro de la ingeniería del software, la Ingeniería Inversa aplicada para llevar a cabo procesos de Reingeniería se ha utilizado comúnmente en la migración de sistemas a nuevas plataformas o tecnologías, como una forma de mejorar el mantenimiento de dichos programas. Una de las grandes dificultades a la que se enfrenta la Reingeniería es la falta de información de alto nivel de los programas construidos que permita un proceso automático de Ingeniería Inversa. Es por esto que la mayoría de las propuestas existentes permiten, sin modificar el software inicial, extraer algún tipo de información del dominio.

Esto se lleva normalmente a cabo mediante técnicas manuales basadas en observar cómo opera la aplicación en sí para producir un nuevo diseño que mejore al anterior o lo adapte a la tecnología actual.

El desarrollo de interfaces de usuario es un proceso que se presta igualmente a la Ingeniería Inversa, debido a que la mayoría de las migraciones se realizan por cuestiones de adaptación del propio software a nuevas librerías gráficas, lenguajes o incluso plataformas de acceso.

2.2.2 Ingeniería Inversa de Base de Datos

Ya que es muy habitual que en los mercados actuales aparezcan nuevos entornos, nuevos sistemas de gestión de datos, más eficaces y mejores que los anteriores, y ciertamente no es cosa sencilla ir cambiando, pues el hacer un traslado de un sistema de gestión de Bases de Datos a otros, no es una tarea trivial. Además, a veces surgen necesidades que no se llevan a cabo por la dificultad que tendría la modificación de aplicaciones, modelos de datos, etc. Un

porcentaje muy elevado de esas mejoras, está muy relacionada con la forma de almacenar los datos. Es un proceso muy costoso tener que desarrollar nuevo software y tener que adaptar todos los datos que se encuentran en un determinado sistema de gestión de Bases de Datos a otro distinto, cuando prácticamente toda la información va a ser la misma. Por tanto, reutilizar lo ya construido aparece como un importante activo a aprovechar.

El proceso de Reingeniería de Bases de Datos, consiste en la recuperación mediante distintos métodos de toda la información de las distintas vistas (física, conceptual y lógica) de la Base de Datos actual o Base de Datos legada (LBD) para en posteriores etapas conseguir, modificar y rediseñar el esquema conceptual, transformando la Base de Datos anterior (LDB) en otra Base de Datos o Base de datos nueva (NBD). Este proceso conlleva entre otros aspectos de vital importancia, la migración de datos de la LDB a la NDB. Por lo tanto, la Ingeniería Inversa una etapa dentro de una metodología general de Reingeniería de sistemas de información, para conseguir especificaciones que permitan un alto grado de satisfacción y obtener un sistema de información basado en un sistema anterior.

2.2.3 Nivel de abstracción

El nivel de abstracción de un proceso de Ingeniería Inversa y las herramientas que se utilizan para realizarlo aluden a la sofisticación de la información de diseño que se puede extraer del código fuente. El nivel de abstracción ideal deberá ser el más alto posible, es decir, el proceso de Ingeniería Inversa debe ser capaz de derivar:

- Su representación de diseño de procesamiento (con un bajo nivel de abstracción).
- La información de las estructuras de datos y de programas (un nivel de abstracción ligeramente más elevado).
- Modelos de flujos de datos y de control (un nivel de abstracción relativamente alto).
- Modelos de entidades y relaciones (un elevado nivel de abstracción)

2.2.4 Métodos de Ingeniería Inversa de Base de Datos Relacionales

Desde los años 80ôs, una amplia gama de los métodos de Ingeniería Inversa de Base de Datos o Database Reverse Engineering (DBRE) se han publicado. Todos consisten en extraer del esquema conceptual de un sistema operacional de la Base de Datos heredada.

Los primeros esfuerzos de la investigación se centraron en transformar un esquema emparentado con llaves primarias conocidas y llaves foráneas a un esquema conceptual. Estos acercamientos utilizaron la información sobre las tablas, los nombres de la columna y las llaves primarias. No especificaron las

fuentes de la información (se da en el esquema físico) y tenían requisitos previos pensados en el esquema físico.

2.2.4.1 Método de Navathe

El método requiere, como entrada, las relaciones en la tercera forma normal y los atributos que se utilizan en más de una relación y deben tener el mismo nombre a través del esquema. La salida es un esquema conceptual expresado en una versión del modelo de la Entidad-Relación, llamada el modelo de Entidad-Categoría-Relacion (ECR), que introduce los conceptos de subclases y de jerarquías de la generalización. La metodología clasifica relaciones y atributos.

2.2.4.2 Método Chiang

Requiere, como entrada, una Base de Datos poblada. Las relaciones están en la tercera forma normal y los atributos de la llave con el mismo dominio deben tener el mismo nombre en todas las tablas, puesto que los nombres de los atributos dominantes se utilizan para deducir referencias entre las tablas. Si no hay una estructura en tercera forma normal que no puede representar más de un tipo de la entidad, debe ser descompuesta en las estructuras que modelan detalles manualmente. La Base de Datos de la entrada contiene cualquier caso erróneo de los datos en sus atributos dominantes. La salida es un modelo extendido de la entidad-relación (EER) con jerarquías de la generalización.

La metodología tiene tres pasos importantes. Primer clasifica las relaciones y los atributos, basados en el esquema relacional y sus llaves primarias. En el segundo paso, se buscan las dependencias de las referencias. Las dependencias posibles se detectan basadas en los identificadores y los nombres de los atributos. Entonces, estas dependencias de la inclusión son validadas haciendo búsquedas en a Base de Datos. Finalmente, se identifican los componentes de ER usando una lista de reglas.

2.2.4.3 Método Premerlani

El método requiere, como entrada, el diccionario de los datos de la fuente del idioma descriptivo de datos, los datos del uso y el analista debe saber la semántica que se usa. No se hace ningunas asunciones en la relación y hace frente a la optimización del diseño y a la mala puesta en práctica del diseño.

Se utilizan las llaves candidatas en vez de las llaves primarias. Las llaves candidatas se identifican con un análisis de índices únicos, exploración automática de datos y del conocimiento del usuario. En un sistema moderno, las llaves extranjeras pueden estar presentes en el DBMS. Si no, las llaves extranjeras pueden ser deducidas investigando tipos de los nombres que emparejan, del dominio y de datos. Una jerarquía de la generalización puede complicar la identificación de llaves extranjeras, esto como salida.

2.2.4.4 Método Petit

Requiere, como entrada, el esquema relacional y los datos como ejemplo, así como código (búsquedas en SQL). La salida es un modelo extendido de la entidad-relación (ER). La metodología analiza condiciones en preguntas y opiniones para recuperar las llaves extranjeras y las dependencias funcionales. Así el esquema se reestructura para obtener un esquema lógico en la tercera forma normal. Finalmente el esquema se traduce a un esquema conceptual y un experto del dominio valida este esquema.

2.3 Metodología genérica de la Ingeniería Inversa de Bases de Datos

El método que será descrito es un tipo genérico que no especifica un tipo de Base de Datos en particular, es llamado DB-MAIN y fue desarrollado Jean-Luc Hainaut.

El método de DB-MAIN requiere, como entrada, toda la información sobre la Base de Datos heredada tal como el DDL, el código fuente, los datos, la documentación, etc. Produce dos salidas. El esquema lógico, que es el esquema que el programador debe entender para poder modificar la Base de Datos heredada y los programas que modifican los datos. La segunda salida es el esquema conceptual como esquema de la Entidad-Relación.

Este método esta divido en tres procesos que dan como resultado dos esquemas, el esquema lógico y el esquema conceptual. En la siguiente figura se muestra los pasos de esta metodología.

2.3.1 Proyecto de preparación

El proyecto de preparación es un paso preliminar, ya que se refiere a la evaluación de los componentes que serán analizados, la evaluación de los recursos necesarios y la definición de la implementación de las operaciones. Aunque este paso no es estrictamente una tarea de la Ingeniería Inversa, si es una forma de administrar el proyecto e identificar la información relevante. Tampoco es una tarea fácil, ya que de ella depende el desarrollo del proyecto. Esta etapa consta de los siguientes procesos:

- Identificación de los componentes y su calidad. Los archivos, programas, reportes, formularios, diccionario de datos, repositorios, programas fuentes, datos y documentación son identificados y evaluados, no todos los componentes tienen la misma calidad y cantidad de información.
- Recuperación de la arquitectura .Consiste en dibujar el procedimiento principal y la constitución de datos del sistema y sus relaciones.

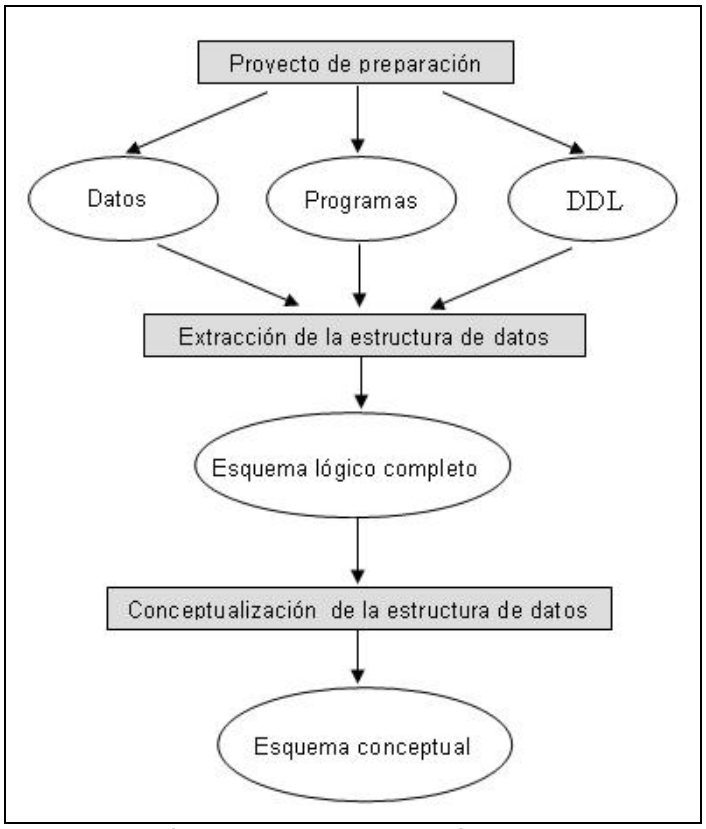


Figura 2-3 Metodología general de la Ingeniería Inversa de Base de Datos

- Definición del proyecto. Define la parte de la aplicación que será analizada y cuáles serán los resultados esperados. Usualmente solo una parte de la aplicación se le puede aplicar Ingeniería Inversa. Esto es importante para definir las limitaciones de las aplicaciones analizadas. En este momento se le puede informar al cliente cuales son las expectativas y así evitar sorpresas.
- Identificación de recursos. Consiste en evaluar los recursos necesitados en términos de técnicas, calendarios, herramientas y presupuesto.
- Planeación de operaciones. Cada paso del proyecto, desde la fecha de termino hasta el presupuesto son dados en esta etapa. Es importante organizar el cronograma de eventos así como definir las personas que dirigirán el proyecto para que en caso de alguna dificultad, la planeación pueda ser ajustada.

2.3.2 Extracción de la estructura de datos

Esta fase consiste en la recuperación del esquema lógico completo, incluyendo todas los estructuras implícitas y explícitas y las restricciones. Una construcción explícita es un componente o una propiedad de una estructura de datos que es declarada a través de una sentencia especifica DDL. Una construcción implícita es un componente o propiedad que se guarda en una estructura de datos, pero que no ha sido declarada explícitamente. En general, el DBMS no detecta las

construcciones implícitas, aunque estas pueden contribuir para una administración a través de triggers[6] por ejemplo.

Este esquema puede ser descrito como la vista de la Base de Datos del programador que tiene o debe tener para desarrollar nuevas aplicaciones y poder darle mantenimiento correctamente. Así, este esquema tiene una descripción de todas las colecciones, los tipos de entidades y atributos de la Base de Datos, con sus nombres físicos y todos las restricciones, implementadas o no.

Ciertamente los sistemas de Base de Datos suministran, en forma legible y procesable, una descripción de este esquema (diccionario de datos, DDL, etc). Sin embargo, la información esencial puede estar perdida en este esquema y otros componentes de la aplicación como vistas, subesquemas, pantallas y diseño de reportes, procedimientos así como fragmentos de documentación son una fuente para poder guiarse. El análisis de cada programa fuente provee una vista parcial de la colección y el tipo de estructuras de las entidades únicamente.

A su vez, la extracción de la estructura de datos se divide en:

2.3.2.1 Análisis del código DDL

El análisis de DDL consiste en analizar la declaración de las estructura de datos, específicamente un DDL, incluyendo en este esquema la creación de scripts y/o declaración de aplicaciones. Esto produce un esquema físico de cada construcción de DDL.

Este proceso es el más simple de la extracción de la estructura de datos. El lenguaje de la mayoría de los sistemas de DBMS es fácil. El modelo de abstracción física proporciona abundantes características. En cada DDL, se incluye las cláusulas previstas para declarar conceptos físicos como índices o clusters, así como conceptos lógicos como entidades o atributos. La separación de las construcciones del DBMS en los niveles estándares de la abstracción puede ser difícil. En la tabla 2-1 se muestran las reglas principales de la abstracción para convertir SQL en una abstracción física.

2.3.2.2 Integración física

Cuando se tiene más de un código fuente de DDL, el analista se encuentra con el problema de tener varios esquemas extraídos, esto ocurre cuando el DBMS no proporciona un diccionario único de datos. Sin embargo, todos los esquemas tienen algunos elementos comunes y cada uno puede incluir los elementos específicos no presentes en el otro. El esquema lógico final debe incluir las especificaciones de todas las visiones parciales.

[6] Es un evento que se ejecuta cuando se cumple una condición establecida al realizar una operación de inserción (insert), actualización (update) o borrado (delete).

2.3.2.3 Refinamiento del esquema

El proceso del refinamiento del esquema es una tarea compleja con la cual varias fuentes de información se buscan para la evidencia de construcciones implícitas o perdidas. El esquema físico explícito obtenido hasta ahora se enriquece con estas construcciones, así conduciendo al esquema físico completo.

Sentencias SQL	**Abstracción física**
create dbspace Espacio	Colección de Espacio
create table Empleados(. . .)in Espacio	La entidad Empleados es de la colección de Espacio
codigo NUMBER(4),	El atributo codigo es numérico con un tamaño de 4
nombre VARCHAR(50),	El atributo nombre es carácter con un tamaño de 25
puesto VARCHAR(25) NOT NULL,	El atributo puesto es obligatorio
supervisor NUMBER(2),	El atributo supervisor es numérico de tamaño 25
fecha_contratacion DATE,	El atributo fecha_contratacion es de tipo fecha
salario NUMBER(6),	El atributo salario es de tipo numérico de tamaño 6
comision NUMBER(4),	El atributo comision es de tipo numérico de tamaño 4
num_depto NUMBER(2)	El atributo num_depto es de tipo numérico de tamaño 2
PRIMARY KEY (codigo)	El atributo codigo es un identificador primario
FOREIGN KEY (num_depto) REFERENCES DEPARTAMENTO	El atributo num_depto hace referencia a la entidad DEPARTAMENTO

Tabla 2-1 Conversión de SQL a la abstracción física

Es importante tener en mente que este análisis es independiente al DBMS.

1. Localizar la estructura de tipo de registros y campos. Un campo declarado como atómico, tiene una descomposición implícita o es la concatenación de campos independientes contiguos. El problema es recuperar la estructura exacta de este campo o este tipo de registro.
2. Localizar campos nulos. La teoría es que cada campo tiene un valor. En general, encontrar un campo sin valor consiste en encontrar un valor especial, puede ser interpretado como un valor conocido o desconocido.
3. Localizar celdas agregadas. Un secuencia de campos aparentemente independiente (direc_numero,direc_calle) fueron originados de un campo compuesto (direccion) el cual puede descomponerse. El problema es para reconstruir este campo compuesto.
4. Localizar campos multivaluados. Un campo, declarado con un solo tipo de valor, aparece como la concatenación de los valores de campos multivaluados.
5. Se puede tener identificadores implícitos, por ejemplo, si tenemos un atributo cliente que no puede tener dos órdenes del mismo producto.
6. Localizar llaves foráneas. Identificar aquellas llaves que dependan de otra tabla.
7. Localizar constraints. Se necesita ubicar las restricciones que cada dato tiene.

8. Localizar cardinalidad y tipo de relación.
9. Localizar redundancias.

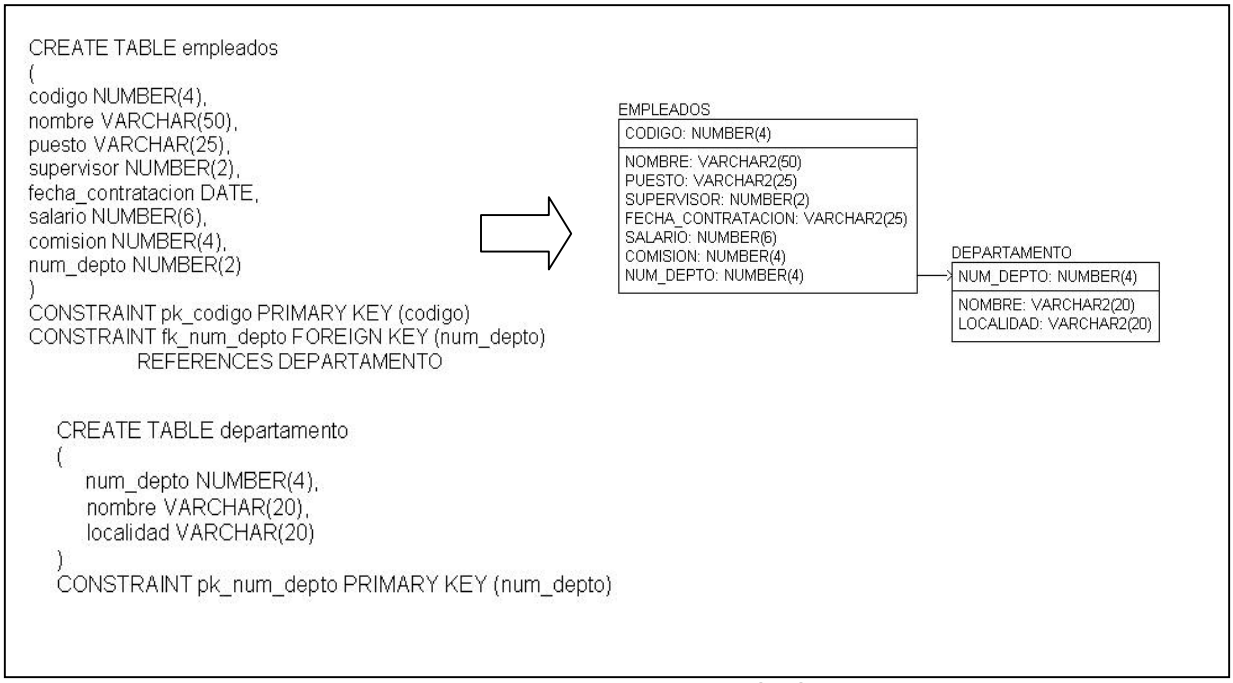


Figura 2-4 SQL-DDL y su abstracción física

2.3.2.4 Limpieza del esquema

Este proceso transforma el esquema físico completo en el esquema lógico. Todas las construcciones físicas pueden ser desechadas a este punto porque no proporcionan ninguna información sobre la estructura lógica de la Base de Datos. Eran útiles por razones técnicas tal como optimización del funcionamiento de la Base de Datos. En esta fase se realiza un estudio del código fuente existente, para comprobar que las restricciones, forma de procesar los datos, significado, etc. se corresponde con el estudio realizado en las fases anteriores. Véase la Figura 2-4.

2.3.3 Abstracción Conceptual

En esta fase se extrae el esquema conceptual a partir del esquema lógico (Figura 2-5), también se le denomina interpretación de las estructuras de datos. Esta fase se divide en tres etapas:

2.3.3.1 Preparación del esquema

El esquema todavía incluye algunas construcciones, tales como archivos y llaves de acceso, que pueden haber sido útiles en la fase de la extracción de la estructura de datos, pero que puede ahora ser desechado. Además, traducir

nombres para hacerlos más significativos y la reestructuración de algunas partes del esquema pueden probar útil antes de intentar interpretarlas.

2.3.3.2 Conceptualización básica

El objetivo principal de este proceso es extraer todos los conceptos semánticos relevantes del esquema lógico. Se presentan en esta etapa dos problemas, requiriendo diversos razonamientos y métodos, son llamados:

- Traslación del esquema. El esquema lógico es la traducción técnica de construcciones conceptuales. Con este proceso el analista identifica los rastros de tales traducciones y las sustituye por su construcción conceptual original. Los modelos de datos pueden compartir un subconjunto importante de traducciones, es decir, a menudo las reglas de traducción que se utilizan en un modelo se utilizaran en otros.
- Esquema optimizado. El esquema lógico se busca para rastros de las construcciones diseñadas para los propósitos de la optimización.

2.3.3.3 Normalización

Este proceso reestructura el esquema conceptual básico para darle las calidades deseadas que se espera de cualquier esquema conceptual. El objetivo es representar la información con una metodología estándar que sea altamente legible.

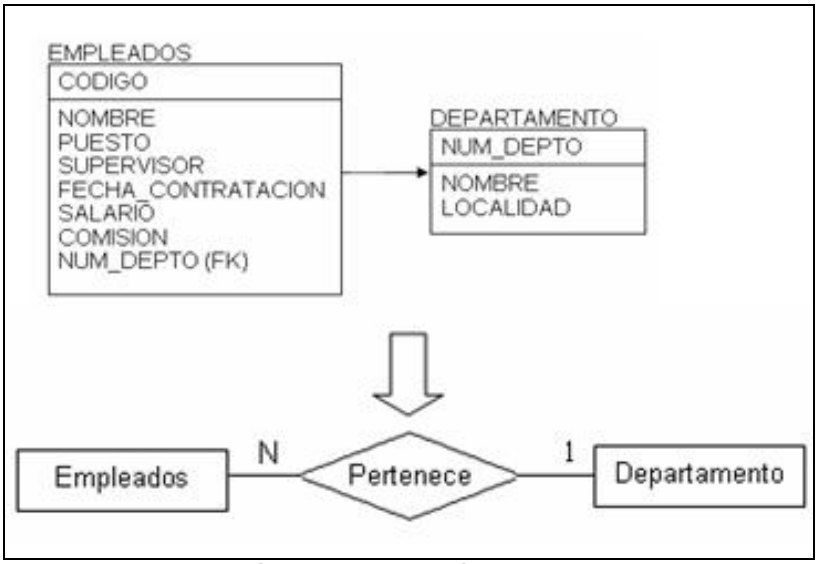


Figura 2-5 Traducción del esquema lógico al esquema conceptual

2.4 Dificultades al aplicar Ingeniería Inversa de Base de Datos

La mayoría de los problemas que se presentan durante el proceso de Ingeniería Inversa se mencionan a continuación:

1. Estructuras implícitas: algunas estructuras no fueron declaradas en el momento de su diseño de forma explícita en la especificación DDL de la Base de Datos, sino que para completar la semántica de la Base de Datos se han utilizado triggers, checks, procedimientos, etc., por lo que para encontrar características como la integridad no basta con observar las definiciones de las tablas (DDL).
2. Diseño pobre o construcciones poco legibles: no todas las Bases de Datos son perfectas aunque funcionen correctamente, a veces se puede encontrar estructuras pobres o incluso erróneas, redundantes o simplemente poco legibles al tener nombres en los campos que son poco clarificadores, etc.
3. Poca documentación: se realizó poca o no existe documentación sobre el diseño.
4. Migrar los datos desde modelos de datos antiguos al modelo relacional. Puede causar que los datos sean inconsistentes o se pierdan.
5. Cambiar el sistema de gestión de Base de Datos.
6. Detección de errores de integridad y persistencia.

Actualmente existen herramientas informáticas que colaboran con la Ingeniería Inversa de Base de Datos, y aunque solo traducen el código para extraer estructuras explícitas, son de gran utilidad mientras se lleva a cabo el proceso. Es por ello que en el siguiente capítulo se mostrarán algunas de estas herramientas existentes en el mercado.

Herramientas de la Ingeniería Inversa de Base de Datos

Desde los comienzos del procesamiento de datos se han buscado métodos, utilidades y técnicas que puedan ayudar a la automatización del ciclo de vida del desarrollo de sistemas de información, completamente o en alguna de sus fases.

La información es fundamental para las operaciones y actividades en las empresas, es necesario manejarla de una forma adecuada y a la vez rápida. Si una empresa quiere competir en el mercado donde se desarrolla necesita contar con sistemas que le provean la información necesaria para realizar sus operaciones en una forma rápida y eficiente si es que quiere tener competitividad. Las utilidades que una organización logra obtener son en proporción a una adecuada organización y la calidad de sus datos. Las empresas que cuentan con una base datos de calidad son aquellas que tienen una cultura organizacional y flexible, es decir, son capaces de adaptarse al cambio tecnológico para el bien de la empresa.

Para la Ingeniería Inversa de Base de Datos es importante tener herramientas que ayuden a cada fase de este proceso para obtener mejores resultados y más eficientes.

3.1 Herramientas CASE

CASE son las siglas en ingles de Computer-Aided Systems Engineering, siendo en español Ingeniería de Sistemas Asistido por Computadora. Las herramientas CASE son un conjunto de métodos, utilidades y técnicas que facilitan la automatización del ciclo de vida del desarrollo de sistemas de información, completamente o en alguna de sus fases, con esto se logra una asistencia a analistas, ingenieros de software y desarrolladores durante dicho ciclo.

La realización de un nuevo software requiere que las tareas sean organizadas y completadas en forma correcta y eficiente. Las Herramientas CASE fueron desarrolladas para automatizar esos procesos y facilitar las tareas de coordinación de los eventos que necesitan ser mejorados en el ciclo de desarrollo de software.

Esta tecnología surge a mediados de los años setenta, cuando empiezan a aparecer las primeras metodologías estructuradas y se inician las investigaciones sobre entornos de desarrollo. En un inicio surgió un simple procesador de palabras que fue usado para crear y manipular documentación y poco a poco se fue introduciendo técnicas graficas y diagramas de flujo de estructuras de datos. Con respecto a esto, el diseño y especificaciones en forma gráfica han sido extremadamente complejos y consumían mucho tiempo para realizar cambios.

Se reemplazaron los paquetes gráficos por paquetes especializados que habilitan la edición, actualización e impresión en múltiples versiones de diseño. Eventualmente, las herramientas gráficas integradas con diccionarios de Base de Datos para producir poderosos diseños y desarrollar herramientas, podrían sostener ciclos completos de diseño de documentos.

Inclusive, la verificación de errores y generadores de casos de pruebas fueron agregados para validar el diseño del software. Todos estos procesos pueden saberse integrados en una simple herramienta CASE que soporta todo el ciclo de desarrollo.

La mejor razón para la creación de estas herramientas fue el incremento en la velocidad de desarrollo de los sistemas. Por esto, las compañías pudieron desarrollar sistemas sin encarar el problema de tener cambios en las necesidades del negocio, antes de finalizar el proceso de desarrollo. Las herramientas CASE también permiten a los analista tener más tiempo para el análisis y diseño y minimizar el tiempo para codificar y probar.

También permite a las compañías competir más efectivamente usando estos sistemas desarrollados nuevamente para compararlos con sus necesidades de negocio actuales. En un mercado altamente competitivo, esto puede hacer la diferencia entre el éxito y el fracaso.

La introducción de las herramientas CASE integradas han comenzado a tener un impacto significativo en los negocios y sistemas de información de las organizaciones. Con un CASE integrado, las organizaciones pueden desarrollar rápidamente sistemas de mejor calidad para soportar procesos críticos del negocio y asistir en el desarrollo y promoción intensiva de la información de productos y servicios.

3.1.1 Objetivos de las Herramientas CASE

Estas herramientas pueden proveer muchos beneficios en todas las etapas del proceso de desarrollo de software, algunas de ellas son:

- Aumentar la productividad, tanto en las áreas desarrollo como en las de mantenimiento de estos sistemas.
- Mejorar la calidad del software desarrollado.
- Reducir tiempos y costos de desarrollo y mantenimiento del software.
- Mejorar la gestión y dominio sobre el proyecto en cuanto a su planificación, ejecución y control.
- Aumentar la biblioteca de conocimiento informático de una empresa ayudando a la búsqueda de soluciones para los requisitos.
- Automatizar el desarrollo del software, la creación de documentación, generación de código, pruebas de errores, herencias y dependencias, gestión de interfaces, creación de flujos y casos de uso.
- Proporcionar una fácil comprensión a través de diagramas, la reutilización del software, la portabilidad y estandarización de la documentación y la utilización y estandarización de distintas metodologías.

- Mejorar el archivo de datos (enciclopedia[1]) de conocimientos y sus facilidades de uso, reduciendo la dependencia de analistas y programadores.
- Una gestión global en todas las fases de desarrollo de software con una misma herramienta.

La principal ventaja de la utilización de una herramienta CASE, es la mejora de la calidad de los desarrollos realizados y, en segundo término, el aumento de la productividad. Para conseguir estos dos objetivos es conveniente contar con una organización y una metodología de trabajo, además de la propia herramienta.

La mejora de calidad se consigue reduciendo sustancialmente muchos de los problemas de análisis y diseño, esenciales en los proyectos de mediano y gran tamaño. La mejora de productividad se consigue a través de la automatización de determinadas tareas, como la generación de código y la reutilización de objetos o módulos.

3.1.2 Clasificación de las Herramientas CASE

Hay diferentes enfoques para clasificar a las herramientas CASE, En primer lugar se suele clasificar, atendiendo a la fase del ciclo de vida que soportan:

- Herramientas de alto nivel U-CASE (Upper CASE o CASE Superior), orientadas a la automatización y soporte de las actividades desarrolladas durante las primeras fases del desarrollo: planificación estratégica, requerimientos de desarrollo.
- Herramientas de nivel medio (Middle CASE o Case Medio),que abarca las fases de análisis y diseño.
- Herramientas de bajo nivel L-CASE (Lower CASE o CASE inferior), que dirigidas a las últimas fases del desarrollo: construcción e implementación.
- Herramientas integradas, I-CASE (integrated CASE o CASE integrado), que son las herramientas que engloban ambos aspectos, es decir abarcan todas las fases del ciclo de vida del desarrollo de sistemas. Son llamadas también CASE workbench[2].
- Juegos de herramientas o Tools-Case, son el tipo más simple de herramientas CASE. Automatizan una fase dentro del ciclo de vida. Dentro de este grupo se encontrarían las herramientas de Reingeniería, orientadas a la fase de mantenimiento.

[1] En el contexto CASE, se entiende por enciclopedia a la Base de Datos que contiene todas las informaciones relacionadas con las especificaciones, análisis y diseño del software.
[2] Son conjuntos integrados de herramientas que dan soporte a la automatización del proceso completo de desarrollo del sistema informático. Permiten cubrir el ciclo de vida completo. El producto final aportado por ellas es un sistema en código ejecutable y su documentación.

Otra posible clasificación, utilizando la funcionalidad como criterio principal, es la siguiente:

- Herramientas de análisis y diseño. Permiten al desarrollador crear un modelo del sistema que se va a construir y también la evaluación de la validez y la consistencia de este modelo. Proporcionan un grado de confianza en la representación del análisis y ayudan a eliminar errores con anticipación. Dentro de esta clasificación destacan las herramientas que permiten crear y modificar diagramas Entidad/Relación, diagramas de flujo de datos, etc. Se tiene herramientas para hacer prototipos como diseñadores de pantallas y generadores de menús. La herramienta tiene la capacidad de análisis de verificación de especificaciones, no sólo sintáctica sino también semántica, por ejemplo, la normalización de un diagrama de datos hasta la tercera forma normal. Se opera sobre un repositorio donde se va almacenando la información necesaria para el funcionamiento de la misma herramienta.
- Herramientas de planificación de sistemas de gestión. Crea modelos sobre los requisitos de información estratégica de una organización. Ayuda a comprender el manejo de la información entre las distintas unidades organizativas. Estas herramientas proporcionan una ayuda importante cuando se diseñan nuevas estrategias para los sistemas de información y cuando los métodos y sistemas actuales no satisfacen las necesidades de la organización.
- Herramientas de gestión de proyectos. La mayoría de estas herramientas se centran en un elemento específico de la gestión de proyecto. Si se utiliza un conjunto de estas herramientas se puede realizar estimaciones de esfuerzo, costo y duración, hacer un seguimiento continuo del proyecto, estimar la productividad y la calidad. Hay herramientas que permite al usuario comprador del desarrollo, hacer el seguimiento que va desde los requisitos, durante el desarrollo hasta el producto final. Aquí se incluyen las siguientes tipos de herramientas.
  - Herramientas de planificación de proyectos.
  - Herramientas de seguimiento de requisitos.
  - Herramientas de gestión y medida.
- Herramientas de prueba. Las herramientas de prueba se conocen también por las siglas CAST (Computer Aided Software Testing), y es un área bastante reciente dentro de la tecnología CASE.
- Herramientas de soporte. Se engloban en esta categoría las herramientas que recogen las actividades aplicables en todo el proceso de desarrollo como documentación, software de sistemas, control de calidad y Base de Datos.
- Generación de código y documentación. A partir de las especificaciones del diseño se puede generar código tanto para los programas como sentencias de definición en SQL. Las herramientas CASE también soportan la creación automatizada de documentación (obtenido a partir de la información almacenada en el repositorio).

- Herramientas de mantenimiento: La categoría de herramientas de mantenimiento se puede subdividir en:
  - Herramientas de Ingeniería Inversa.
  - Herramientas de reestructuración y análisis de código.
  - Herramientas de Reingeniería.

3.1.3 Herramientas CASE de Ingeniería Inversa

Con estas herramientas se lleva a cabo:

- Ingeniería Inversa de datos. Cuentan con la capacidad de extraer la información del código fuente que describe la estructura de los elementos de datos, construyendo así diagramas E/R partiendo de esquemas relacionales, en red o, incluso, ficheros.
- Ingeniería Inversa de procesos. Permiten aislar la lógica de las entidades, y las reglas del negocio a partir del código.
- Reestructuración de código fuente. Modifican su formato o implantan un formato estándar.
- Redocumentación. Genera diagramas a fin de lograr una mejor compresión del código.
- Análisis de código. Sus funcionalidades van desde la identación automática del código fuente hasta la posibilidad de ir visualizando dinámicamente las llamadas del mismo.

3.1.4 Componentes de las Herramientas CASE

Las herramientas CASE se componen de:

3.1.4.1 Repositorio o depósito de datos

Es la Base de Datos de información central de la herramienta CASE, en este depósito se almacena toda la información creada y generada a través de las etapas que cubre la herramienta de determinado proceso de desarrollo, como por ejemplo, componentes de análisis y diseño. También se le denomina Diccionario de Recursos de Información.

La mayoría de herramientas CASE poseen un repositorio propio o bien trabajan sobre un repositorio suministrado por otro fabricante o vendedor. Apoyándose en la existencia del repositorio se efectúan comprobaciones de integridad y consistencia como:

- La existencia de datos no definidos.
- La existencia de datos autodefinidos[3].
- Los alias[4] deben ser correctos y estén actualizados.

[3] Datos que se emplean en una definición pero que no han sido definidos previamente.
[4] Referencias a un mismo dato empleando nombres distintos.

- Las características más importantes de un repositorio son:
  - Tipo de información. Que contiene alguna metodología concreta, datos, gráficos, procesos, informes, modelos o reglas.
  - Tipo de controles. Si incorpora algún módulo de gestión de cambios, de mantenimiento de versiones, de acceso por clave, de redundancia de la información.

3.1.4.2 Módulo para creación y modelado

Consiste en dar soporte para la creación de los diagramas más utilizados para el análisis y diseño del software como los diagramas de flujo de datos, el modelo Entidad-Relación y los diagramas de clase. Las principales características con las que debe cumplir este modulo y que han sido implementadas son el número máximo de objetos que se permiten en un diagrama y la posibilidad de deshacer los cambios hechos a los diagramas.

Algunas características referentes a los diagramas son:

- Número máximo de niveles para poder soportar diseños complejos.
- Número máximo de objetos que se pueden incluir para no encontrarse limitado en el diseño de grandes aplicaciones.
- Número de diagramas distintos en pantalla o al mismo tiempo en diferentes ventanas.
- Dibujos en formato libre con la finalidad de añadir comentarios, dibujos, información adicional para aclarar algún punto concreto del diseño.
- Actualización del repositorio por cambios en los diagramas. Siempre resulta más fácil modificar de forma gráfica un diseño y que los cambios queden reflejados en el repositorio.
- Control sobre el tamaño, fuente y emplazamiento de los textos en el diagrama.
- Comparaciones entre gráficos de distintas versiones. De esta forma será más fácil identificar qué diferencias existen entre las versiones.
- Inclusión de pseudocódigo, que servirá de base a los programadores para completar el desarrollo de la aplicación.
- Posibilidad de deshacer el último cambio, facilitando que un error no conlleve perder el trabajo realizado.

3.1.4.3 Analizador de sintaxis

El analizador de sintaxis verifica la validez y el correcto uso de la información introducida a la herramienta, de acuerdo a la notación soportada. Este componente es muy importante para ayudar al desarrollador a no cometer errores de sintaxis al momento de crear los diagramas y generar un código más completo y preciso.

3.1.4.4 Generador de código

Es uno de los componentes indispensables de las herramientas CASE. Las características más importantes de este componente son: el o los lenguajes(s) de programación para los que se genera código y el alcance de la generación del cuerpo del programa.

Las características más importantes de los generadores de código son:

- Lenguaje generado. Si se trata de un lenguaje estándar o un lenguaje propietario.
- Portabilidad del código generado. Capacidad para poder ejecutarlo en diferentes plataformas físicas y/o lógicas.
- Generación del esqueleto del programa o del programa completo. Si únicamente genera el esqueleto será necesario completar el resto mediante programación.
- Posibilidad de modificación del código generado. Suele ser necesario acceder directamente al código generado para optimizarlo o completarlo.
- Generación del código asociado a las pantallas e informes de la aplicación. Mediante esta característica se obtendrá la interface de usuario de la aplicación.

3.1.4.5 Módulo generador de documentación

Consiste en generar automáticamente una documentación a partir de datos del repositorio, así como proporcionar una interfaz con otras herramientas como son procesadores de texto, editores gráficos, etc.

3.2 Características de las Herramientas CASE para el soporte de la Ingeniería Inversa

Se pueden identificar las siguientes características con las que debe contar una herramienta para soportar las actividades de la Ingeniería Inversa.

- Las herramientas deben permitir tener una alta flexibilidad de operación de modelado y altamente interactivas.
- Las funciones específicas deben ser fáciles de desarrollar, incluso para un solo uso.
- Las herramientas deben incluir interfaces de *browsing* y *querying* con una gran variedad de fuentes de información. Debe ser adaptable a las funciones para la automatización y asistencia específica de extracción que debe estar disponible para cada uno de ellos.
- Debe proveer un texto enriquecido para el analizador de procesos, incluyendo programas de entendimiento. Puede ser lenguaje independiente, fácil de entender y para programar. Debe acoplarse con las especificaciones de los procedimientos.

- Es muy importante que incluya un set de funciones de ingeniería tanto inversa como directa.
- Debe comunicarse fácilmente con otras herramientas de desarrollo de la organización.
- Las herramientas CASE deben proveer varios formas de visualizar ambos fuentes de texto y abstracción de estructuras.
- Deben proveer técnicas de transformación, en particular para poder revertir las transformaciones comúnmente usadas para el diseño empírico de Base de Datos.
- EL repositorio de las herramientas CASE pueden registrar todos los vínculos de los esquemas a los diferentes niveles de abstracción. Más generalmente, las herramientas pueden asegurar la posibilidad de encontrar el proceso de Ingeniería Inversa.

1.3 Herramientas CASE en el mercado capaces de aplicar Ingeniería Inversa

A continuación se mencionan algunas características de herramientas que pueden realizar Ingeniería Inversa de Base de Datos.

❒ **XCase**

Es una herramienta de tipo CASE que ofrece potentes funciones para la automatización y el mantenimiento del ciclo de vida completo de Bases de Datos. xCase automatiza el trabajo y realiza, de forma transparente, las tareas más tediosas y repetitivas.

xCase dispone de una sofisticada interfaz visual que permite diseñar de forma precisa Bases de Datos capturando todos los detalles de la información que debe manejar. Los mecanismos de automatización y mantenimiento de xCase ayudan a completar el análisis y diseño de las Bases de Datos de manera completa e íntegra. Obsérvese la ilustración 3-1.

❒ **ConceptDraw**

Se puede modelar un diagrama Entidad-Relación optimizada, diseñando relaciones completas en una Base de Datos. También es capaz de generar automáticamente estructuras de Bases de Datos por medio de Ingeniería Inversa. Tienen compatibilidad con múltiples DBMS de destino permitiendo crear un diagrama y diseñar los demás basándolos en ésta. Cuenta con la capacidad de exportación avanzada de un diagrama de diseño de Base de Datos y hace representaciones gráficas flexibles de tipos de datos permitiendo añadir y remover tipos de datos de una DBMS, incluyendo tipos de datos modificados, y crear tipos de datos con auto incremento. Obsérvese la ilustración 3-2.

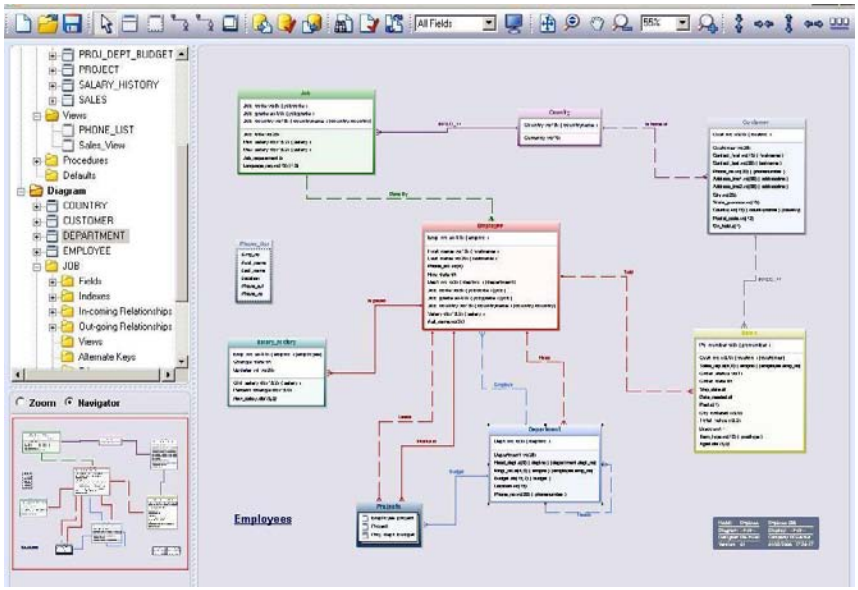


Ilustración 3-1 X-Case

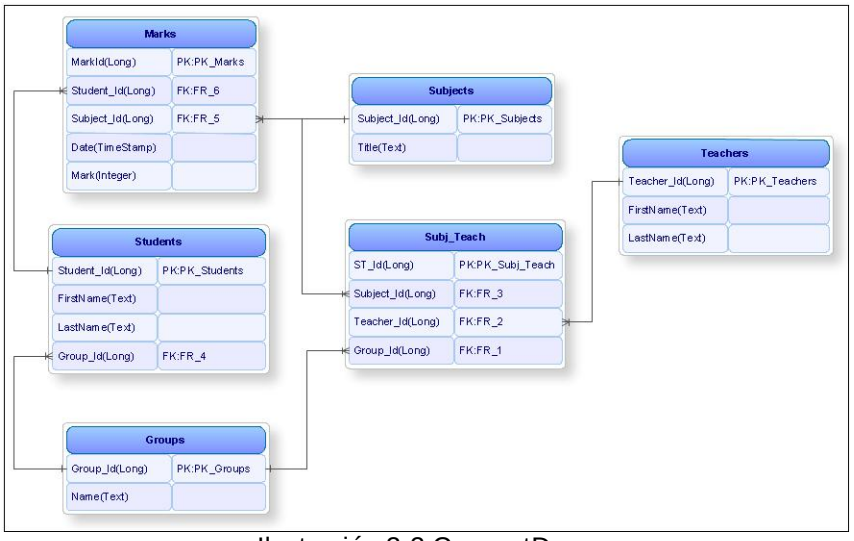


Ilustración 3-2 ConceptDraw

❒ Office Visio

Ofrece soporte para diagramas de Bases de Datos específicos que ofrecen soporte para las técnicas de modelado, las notaciones relacionales, modelos de relación entre objetos, diagramas de niveles de entidades y de esquemas y modelos de datos de productos. Se pueden crear diagramas arrastrando las formas hasta la página de dibujo o aplicando la Ingeniería Inversa a una estructura de Base de Datos antigua desde una serie de Bases de Datos compatibles entre las que se incluyen Microsoft SQL Server, Microsoft Access, Oracle y la DB2 de IBM. Obsérvese la ilustración 3-3.

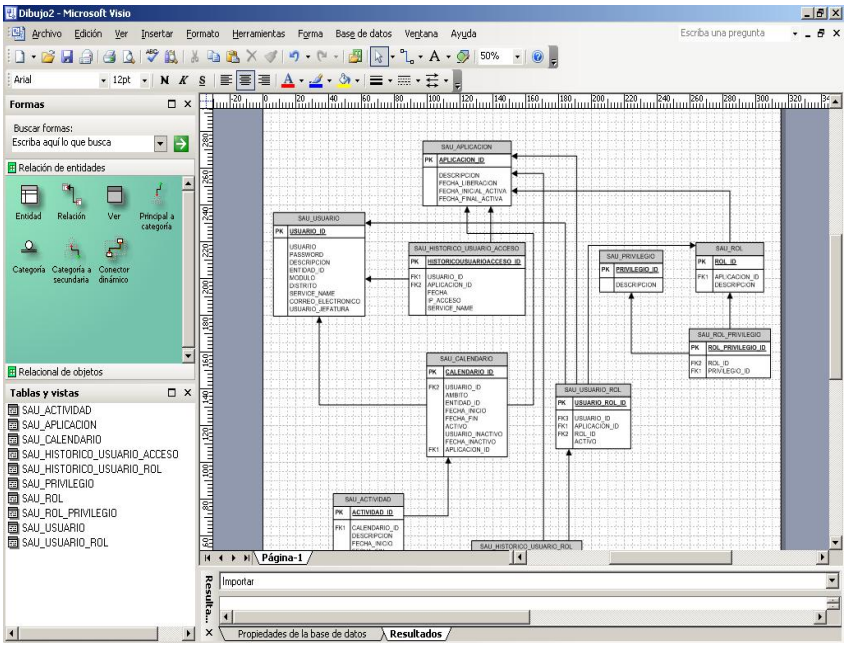


Ilustración 3-3 Office Visio

❒ Semantor

Semantor en una solución avanzada para Ingeniería Inversa de aplicaciones de procesamiento de datos, habilitando a cada uno de sus componentes para tener detalle de una aplicación hasta el nivel más detallado.

Su aplicación directa es:

- Ingeniería de mantenimiento.
- Ingeniería Inversa de aplicaciones y retro-documentación.
- Análisis de impacto.
- Control de calidad y aplicación de estándares de desarrollo.

La calidad de análisis de Semantor y los diferentes puntos de vista proporcionados para un programa (control de flujo, flujo de datos, análisis de calidad, señalamiento de defectos de estructura), y el poder de sus vistas gráficas, están enfocadas como otros varios elementos a obtener o retomar el conocimiento de los programas.

❒ SchemaSpy

Permite hacer Ingeniería Inversa de una Base de Datos para conocer su estructura o detectar incorrecciones. SchemaSpy genera una serie de páginas HTML con gráficos en donde se representa visualmente la estructura de la Base de Datos y las relaciones entre las tablas. Estas páginas son navegables, lo que permite explorar la Base de Datos de un modo intuitivo, ampliar o resumir información, etc. Es una herramienta programada en Java y puede utilizarse con

gran cantidad de Bases de Datos (Oracle, MSQLServer, MySQL, etc). Obsérvese la ilustración 3-4.

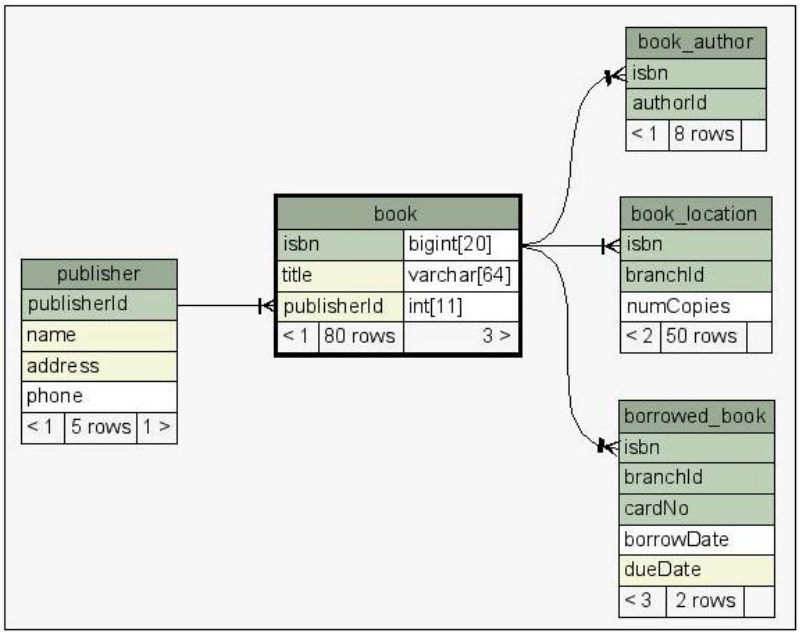


Ilustración 3-4 SchemaSpy

❒ DBDesigner

Es un entorno visual integrado para el diseño, modelado, creación y mantenimiento de Bases de Datos relacionales. Con acabado profesional, el programa dispone de infinidad de opciones y funcionalidades, además de ser bastante intuitivo y fácil de utilizar.

Proporciona facilidades para el diseño, el modelado, la creación y el mantenimiento de Bases de Datos. Realiza Ingeniería Inversa de Bases de Datos MySQL, Oracle, MSSQL y cualquiera con driver ODBC.

Tiene características interesantes como la posibilidad de poder hacer Ingeniería Inversa: generar el modelo Entidad-Relación a raíz de una Base de Datos existente, también permite la sincronización de trabajo con servidores de Bases de Datos (para ir aplicando los últimos cambios que se hace sobre el modelo relacional directamente a la Base de Datos) y dispone de un modo diseño con el que crear y mantener visualmente el modelo de la Base de Datos y el Modo Consulta, completa interfaz con la que se puede trabajar con los datos de las tablas y construir consultas SQL. Además, los datos los guarda con estructura XML[5] pudiendo trabajar con ellos a través de otras aplicaciones. Y, finalmente,

[5] Extensible Markup Language o lenguaje de marcas extensible, permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

dispone de un sistema de plugins[6] de fácil manejo y desarrollo para expandir y personalizar el manejo de esta herramienta. Obsérvese la ilustración 3-5.

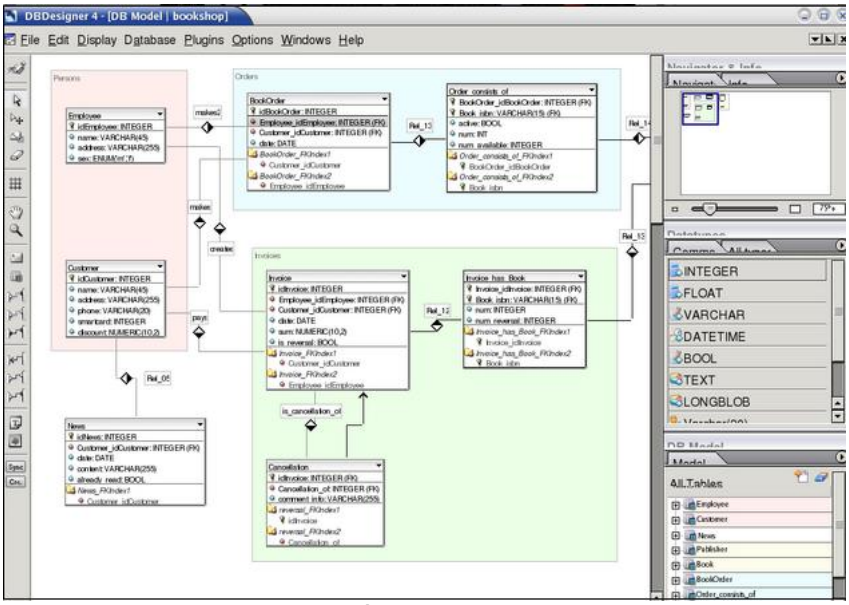


Ilustración 3-5 DBDesigner

❒ ERwin

Es una herramienta de diseño de Base de Datos. Brinda productividad en diseño, generación, y mantenimiento de aplicaciones. Desde un modelo lógico de los requerimientos de información, hasta el modelo físico perfeccionado para las características específicas de la Base de Datos diseñada, ERwin permite visualizar la estructura, los elementos importantes, y optimizar el diseño de la Base de Datos. Genera automáticamente las tablas y miles de líneas de proceso almacenados y triggers para los principales tipos de Base de Datos. Obsérvese la ilustración 3-6.

❒ PowerDesigner

PowerDesigner es una suite de aplicaciones de Powersoft para la construcción, diseño y modelado de datos a través de diversas aplicaciones. Es la herramienta para el análisis, diseño inteligente y construcción sólida de una Base de Datos y un desarrollo orientado a modelos de datos a nivel físico y conceptual, que dan a los desarrolladores Cliente/Servidor la más firme base para aplicaciones de alto rendimiento. Obsérvese la ilustración 3-7.

[6] Es una aplicación informática que interactúa con otra aplicación para aportarle una función o utilidad específica, ésta aplicación adicional es ejecutada por la aplicación principal.

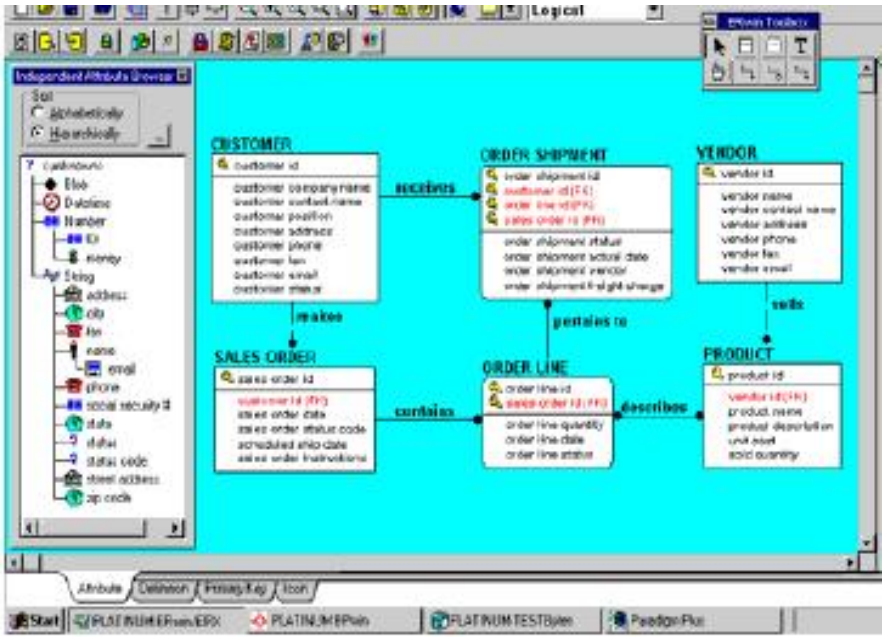


Ilustración 3-6 ERwin

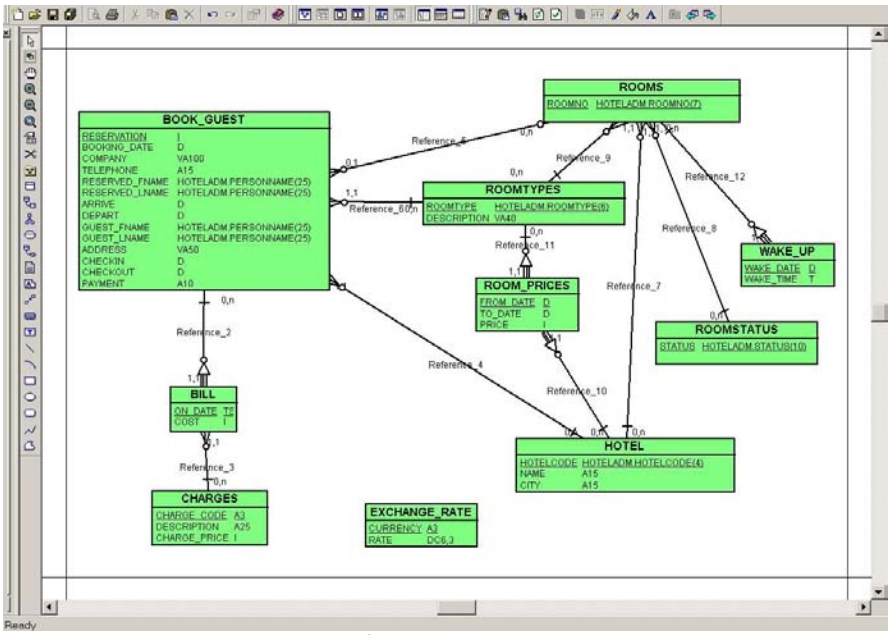


Ilustración 3-7 PowerDesigner

Hasta este punto se han tratado procedimientos para la realización de la Ingeniería Inversa de Base de Datos, pero hay otros aspectos que son necesarios mencionar referente a la seguridad y que serán tratados en el siguiente capítulo.

Seguridad de Base de Datos

La información contenida en una Base de Datos tiene gran valor. Es por esto que se debe buscar procesos y/o actividades que logren garantizar que los datos se encuentren seguros frente a usuarios malignos que traten de leer dicha información valiosa o peor aún, que intenten atacar mediante la manipulación o destrucción de los datos. Así mismo, aunque estos usuarios sean autorizados no se está libre de que, simplemente por ignorancia, cometan errores y afecten la información de la Base de Datos. Los DBMS normalmente disponen de complejos sistemas de permisos a usuarios o grupo de usuarios que restringen la modificación de la Base de Datos o cualquier anomalía, pero también es necesario contar con estrategias que restrinjan el acceso a todos los usuarios no autorizados. Así mismo se debe contar con un plan de contingencia en caso de que la Base de Datos haya sido afectada para así saber las acciones a seguir y recuperar en breve la información.

De esta forma, tanto la estructura de la Base de Datos como la privacidad de los datos también deben ser protegidos ya que es una responsabilidad tanto jurídica como moral.

El concepto de seguridad, se puede entender como la protección de los datos contra acceso, alteración o destrucción no autorizados.

La integridad, confidencialidad y disponibilidad de la información de la Base de Datos son los tres aspectos fundamentales de la seguridad. Se trata de garantizar que los datos almacenados sean veraces y que no sean corruptos, como que solo acceda a la información a los usuarios que les corresponde, así como disponer de los datos cuando sean requeridos a su uso. Esto es fundamental en aplicaciones que deben correr en cualquier horario o en ciertos rangos de horario independientemente del número de accesos simultáneos que se estén realizando.
Para lograr estos objetivos y atender la seguridad en todos sus aspectos se consideran tres áreas:

- Seguridad lógica: Mantener la integridad y consistencia de los datos independiente de aplicaciones, protocolo, procesos informáticos, etc. Asimismo implica restringir el acceso a los datos por parte de usuarios no autorizados o que dichos usuarios solo accedan a lo que les corresponde.
- Seguridad física: Mantener la integridad física de los archivos donde se almacena la Base de Datos y los logs de transacciones, en disco. Se refiere a la protección del hardware.
- Seguridad jurídica: legislación aplicable, la política general, normas, procedimientos, convenciones internas, etc.

Para garantizar el nivel de protección óptimo frente a las posibles amenazas se deberán implementar principalmente y de forma coherente, medidas de seguridad de tres tipos:

- Activas o preventivas. Seguridad lógica como un firewall, la valla en seguridad física y las normas en seguridad político- corporativa.

- Pasivas, como auditoria, detección y/o alarma. Por ejemplo, las cámaras de circuito cerrado los sistemas IDS y los planes de auditoría.
- De recuperación, como por ejemplo el Disaster Recovery Plan, las copias de seguridad o Backup, y la SAI y/o grupo electrógeno.

4.1 Seguridad lógica de las Bases de Datos

El concepto de seguridad es medido por la protección frente a ataques externos o fallos del mismo software, fallos en el equipo donde reside o la manipulación errónea o malintencionada de los usuarios o administradores. La Base de Datos debe ser protegida contra el fuego, el robo y otras formas de destrucción. Pero si la base llega a sufrir problemas, los datos deben poder ser reconstruidos. Debe haber controles sobre los accesos y un monitoreo de acciones sobre los usuarios para descubrir cualquier acción indebida.

Los datos deben poder ser sometidos a procesos de auditoría. La falta de auditoría en los sistemas de computación ha permitido delitos.Para ello, se involucran muchas áreas: ingenieros, diseñadores, operadores, etc. Que deberán trabajar sobre sistemas operativos, Bases de Datos, elementos de red, sistemas de almacenamiento y recursos económicos para lograr un único fin, que el acceso los datos los realicen las personas que están autorizadas para hacerlo y que estos datos puedan recuperarse si se presentan contingencias.

En un primer nivel de control de acceso lógico es necesario implementar tecnología firewall como primer filtro en el que se establecen qué redes de confianza pueden acceder a las Bases de Datos. Posteriormente, es importante analizar la seguridad de las aplicaciones que acceden a la propia Base de Datos, ya que normalmente el usuario no se introduce directamente a la Base de Datos, sino que utiliza un cliente o una aplicación web que es realmente quien se identifica y accede al motor de la Base de Datos. La seguridad del código es necesaria para evitar ataques, originados por la falta de validación de los datos de entrada en la aplicación que accede a la Base de Datos.

4.1.1 Acceso a usuarios

Un usuario es creado por el administrador de la Base de Datos (DBA) y se le asigna una clave de acceso. La creación de usuarios es necesaria para que pueda ser identificado de forma independiente y pueda acceder a lo que necesita. Los usuarios son reconocidos por un ID-usuario[1], único para cada usuario en una instalación. En ocasiones es posible manejar grupos de usuarios identificados por ID-grupo.

Para tener un mayor control sobre los usuarios se han creado mecanismos como los siguientes:

[1] ID significa que es el identificador del campo.

- Permisos de usuario: Se definen las operaciones o acciones permitidas para cada usuario. Los usuarios pueden tener acceso a tablas, campos, procedimientos almacenados, triggers, vistas, etc., y sobre estos se pueden conceder privilegios, según el caso, que permitan consultar, insertar, actualizar, borrar, ejecutar, etc.
- Roles: Son un conjunto de permisos sobre determinados objetos de la Base de Datos. Por ejemplo, el rol de administrador tiene permiso sobre todos los objetos y puede hacer cualquier acción, mientras que un usuario de consulta solo tiene permisos de selección sobre los objetos, pero no puede borrarlos o modificarlos.
- Vistas: Las vistas son una excelente forma de dar al usuario la información que necesita. Son simples consultas a través de las cuales el usuario final únicamente ve determinadas columnas, filas o campos que cumplan un criterio. Se crea así un esquema conceptual a partir de lo que el usuario solicita. Esto evitará tener información redundante. Tienen las siguientes ventajas:
    - Las vistas proporcionan un nivel de seguridad, ya que permiten excluir datos para que ciertos usuarios no los vean.
    - Las vistas proporcionan un mecanismo para que los usuarios vean los datos en el formato que deseen.
    - Una vista representa una imagen consistente y permanente de la Base de Datos, incluso si la Base de Datos cambia su estructura.

Un buen manejo de estas posibilidades permite el control de las actividades de los usuarios sobre el sistema. Este mecanismo es conocido como Control Discriminatorio, el cual posibilita que diferentes usuarios tengan privilegios de acceso a diferentes objetos de Bases de Datos.

En situaciones en que los datos sean críticos, se debe contar con el riesgo de violación de la seguridad por una persona no autorizada, además de errores involuntarios que igual pueden causar inconsistencias o falta de veracidad de la información.

Para estos casos es interesante mantener un archivo de auditoría (conocidos como archivos de logs), donde son registradas todas las operaciones realizadas por los usuarios de las Bases de Datos. En caso de sospecha de falla en la seguridad, este archivo puede ser consultado para conocer los daños causados y/o identificar a los responsables de las operaciones irregulares.

4.1.2 Plan de contingencia

La mejor forma de proteger los datos es a través de un plan de contingencia. Este es definido como una estrategia planificada constituida por un conjunto de recursos de respaldo, una organización de emergencia y unos procedimientos de

actuación encaminada a conseguir una restauración progresiva y ágil de los servicios de negocio afectados por una paralización total o parcial de la capacidad operativa de la empresa.

Esa estrategia es el resultado de todo un proceso de análisis y definiciones que es lo que da lugar a las metodologías y debe plasmarse en un manual.

Un plan de contingencia tiene como objetivos:

- Definir los lineamientos y procedimientos oportunos para responder efectivamente ante una contingencia.
- Brindar un alto nivel de protección contra todo posible evento de efectos negativos sobre el personal, las instalaciones y equipos, la población local y la propiedad privada.
- Reducir la magnitud de los impactos potenciales ambientales y otros impactos durante la fase de ejecución y operación del proyecto.

Es importante estudiar todo lo que pueden paralizar la actividad y producir pérdidas en el negocio. Para lograr un plan de contingencia se desarrollan las siguientes fases:

4.1.2.1 Fase I Evaluación

1. Constitución del grupo de desarrollo del plan. Se asigna un responsable del plan y debe estar formado por líderes de las áreas que se desean cubrir dicho plan y asignar responsables de las áreas afectadas, representantes de auditoría interna y apoyo legar de la organización. Su elaboración ha de desarrollarse con la continua supervisión por parte de la dirección ya que durante la elaboración y/o ejecución de éste, deberán comprometerse recursos y aprobarse procedimientos especiales que requieran un nivel de autorización superior.
2. Identificación de las funciones críticas. Se necesita identificar los elementos de la empresa que puedan ser críticos ante algún desastre y darles prioridad por orden de importancia.
3. Definición y documentación de los posibles escenarios de fallas Se refiere a las posibles fallas que pueden presentarse para cada función crítica. Estas pueden ser:
    - Internas: en este caso la falla está restringida a la empresa. Puede tratarse de problemas informáticos o sistemas automáticos dependientes de fechas (sistemas de acceso, circuitos de ascensores, etc.). También deben incluirse en esta categoría los siniestros que podrían producir incendios, utilización indebida de medios magnéticos de resguardo, o cualquier otro daño de origen físico que pudiera provocar la pérdida masiva de información.
    - De interfaces: se trata de problemas en la información recibida o enviada hacia otras empresas.

- Infraestructura básica: se trata de problemas asociados con la carencia de fuentes de energía, de comunicaciones, abastecimiento de insumos, transporte, etc.

4. Análisis del impacto del desastre en cada función crítica. Consiste en realizar un análisis del impacto de cada problema sobre cada una de las funciones críticas de la organización, teniendo en cuenta las siguientes prioridades:
   - Evitar pérdidas de vida.
   - Satisfacer las necesidades básicas.
   - Reanudar las operaciones lo antes posible.
   - Proteger el medio ambiente.
   - Lograr las conexiones con los principales clientes y proveedores.
   - Mantener la confianza en la empresa.

   Una correcta cuantificación del impacto económico de cada problema ayudará a una correcta selección de la solución alternativa.
5. Definición de los niveles mínimos de servicio. Se trata de definir los mínimos niveles de servicio aceptables para cada problema que se pueda plantear. Es importante que dicho nivel sea aceptado por las áreas afectadas y cuente con el aval de auditoría interna y la intervención de su asesoría legal.
6. Identificación de las alternativas de solución. Se debe identificar las soluciones alternativas para cada uno de los problemas previsibles. Es posible considerar la implementación de procesos manuales y definir tareas críticas por un tiempo determinado.
7. Evaluación de la relación coste/beneficio de cada alternativa. Deberá determinarse la mejor solución desde el punto de vista costo/beneficio para cada proceso crítico y su tiempo de elaboración con un nivel de servicio que satisfaga el nivel mínimo.

4.1.2.2 Fase II Planificación

1. Documentación del plan de contingencia: El contenido debe ser:
   - Objetivo del plan: definir el plan que permitirá continuar la operación de tareas críticas, por ejemplo: continuar las actividades en forma normal, continuar las actividades con un deterioro aceptable de la calidad y productividad o diferir las actividades por razones económicas o de seguridad.
   - Tiempo máximo sin servicio: si es posible cuantificar, el tiempo que la empresa puede sobrevivir con el mínimo de servicio o ninguno.
   - Costos estimados: se refiere a los costos asociados a la solución alternativa.
   - Recursos necesarios: Se debe indicar qué tipo de recursos se necesitan para la solución alternativa: personal, formularios, máquinas, proveedor externo, etc.
   - Criterio disparador: cuál será el evento que comenzará la operatoria alternativa y quién lo activará.

- Roles y responsabilidades: definir qué hará cada persona afectada al plan y cuál es su responsabilidad.
- Capacitación del personal: según la tarea que cada persona cumpla en el plan debe ser capacitada para esta nueva tarea.
- Retorno a la operación normal: deberá ser programada la entrada a la operación normal y, especialmente la recuperación de información dañada o corrompida.
- Comunicaciones: será necesario informar del servicio que se prestará tanto al personal de la Organización como al público usuario.

Se debe considerar otras empresas u organizaciones que sean afectadas por la información de la propia creando interfaces.

2. Validación del plan de contingencia. El plan de contingencia debe ser validado con los responsables de las áreas involucradas y con el área jurídica para verificar que con la implementación del plan no se acarrea problemas legales.

4.1.2.3 Fase III Pruebas y mantenimiento

1. Definir y documentar las pruebas del plan. Es necesario definir las pruebas del plan y el personal y recursos necesarios para su realización. Una correcta documentación ayudará a la hora de realizar las pruebas.
2. Obtener los recursos necesarios para las pruebas. Deben obtenerse los recursos para las pruebas, ya sean recursos físicos o mano de obra para realizarlas.
3. Capacitar al personal que intervendrá en las pruebas. La capacitación del equipo de contingencia y su participación en pruebas son fundamentales para poner en evidencia posibles fallas del plan.
4. Ejecutar las pruebas y documentarlas: Mientras mayores sean los ensayos de su plan, aumentará la certidumbre de que las tareas fundamentales de la empresa. Es necesario documentar las pruebas para su aprobación por parte de las áreas afectadas.
5. Ejecutar y documentar las pruebas de reiniciación del proceso normal.
6. Actualizar el plan de contingencia de acuerdo a los resultados obtenidos en las pruebas. Será necesario retroalimentar el plan de acuerdo a los resultados obtenidos en las pruebas.

Asimismo se define la estrategia de mantenimiento, la organización destinada a ello y la normativa y procedimientos necesarios para llevarlo a cabo.

Al ejecutar el plan en una situación de contingencia no se debe buscar resolver la causa del problema, sino asegurar la continuidad de las tareas críticas de la empresa. Los datos afectados por el siniestro que pudiesen haber quedado corruptos, deben corregirse usando los procedimientos ya definidos. En general, la reiniciación del proceso normal no implica la cancelación del alternativo, salvo que deban utilizarse los mismos recursos. Si esto no es así, durante cierto tiempo, los

procesos deberían ejecutarse en paralelo para asegurar que la reiniciación de la operación normal es correcta y, ante cualquier defecto, continuar con el de contingencia.

El plan de contingencia deberá estar disponible en un lugar visible para que todo el personal pueda acceder a él, así mismo al finalizar cada jornada se deberá evaluar tipos de riesgos que se hubiesen generado durante las actividades, con la finalidad de adaptar y/o complementar las acciones del plan.

Una vez finalizado el plan se debe recoger información del funcionamiento del plan con el fin de evaluarlo y analizar la efectividad del mismo y orientar las recomendaciones sugeridas para efectuar cambios en el mismo.

4.2 Seguridad física de las Bases de Datos

El hardware es frecuentemente el elemento más caro de todo sistema informático. Por tanto, las medidas encaminadas a asegurar su integridad son una parte importante de la seguridad física de cualquier organización

4.2.1 Acceso físico

La posibilidad de acceder físicamente a cualquier sistema operativo hace inútiles casi todas las medidas de seguridad que se hayan aplicado. Es por eso que prevenir dichos accesos es necesario. Los siguientes puntos son una buena forma de prevención.

- Controlar el acceso de personal. Es necesario identificar y autorizar a los usuarios con la implementación de tarjetas de acceso, con el uso de códigos o palabras claves, impresiones digitales, reconocimiento de voz, barrido de retina, etc.
- Controlar el acceso al equipo. Se logra otorgando derechos de acceso dados por la terminal, por la operación que puede realizar o por la hora del día. También puede implementarse técnicas de cifrado para proteger datos en Base de Datos distribuidas o con acceso por red o Internet.

Cuando la prevención es difícil por cualquier motivo ya sea técnico, económico, humano, se debe detectar cuanto antes el ataque, para minimizar así sus efectos. Aunque en la detección de problemas, generalmente accesos físicos no autorizados, intervienen medios técnicos, como cámaras de vigilancia de circuito cerrado o alarmas, en entornos más normales el esfuerzo en detectar estas amenazas se ha de centrar en las personas que utilizan los sistemas y en las que sin utilizarlos están relacionadas de cierta forma con ellos.

4.2.2 Desastres naturales

Un problema que no suele ser tan habitual, pero que en caso de producirse puede acarrear gravísimas consecuencias, es el derivado de los desastres naturales y su falta de prevención.

- Terremotos. Es muy recomendable no situar nunca equipos delicados en superficies muy elevadas ya que los equipos pueden caer o situar equipos cerca de las ventanas: si se produce un temblor pueden caer por ellas.
- Tormentas eléctricas Los equipos deben situarse lo más alejados posible de la estructura metálica de los edificios. Un rayo en el propio edificio, o en un lugar cercano, puede inducir un campo electromagnético lo suficientemente grande como para dañar el hardware o discos.
- Inundaciones y humedad. en ambientes extremadamente secos el nivel de electricidad estática es elevado, lo que, como veremos más tarde, puede transformar un pequeño contacto entre una persona y un circuito, o entre diferentes componentes de una máquina, en un daño irreparable al hardware y a la información. No obstante, niveles de humedad elevados son perjudiciales para los equipos porque pueden producir condensación en los circuitos integrados, lo que origina cortocircuitos que evidentemente tienen efectos negativos sobre cualquier elemento electrónico de una máquina. Para prevenir algún desastre por inundaciones se puede hacer la instalación de un falso suelo por encima del suelo real, o simplemente tener la precaución de situar a los equipos con una cierta elevación respecto al suelo, pero sin llegar a situarlos muy altos por los problemas que se comentaron al hablar de terremotos.
- Incendios. Una causa casi siempre relacionada con la electricidad son los incendios, y con ellos el humo; aunque la causa de un fuego puede ser un desastre natural, lo habitual en muchos entornos es que el mayor peligro de incendio provenga de problemas eléctricos por la sobrecarga de la red debido al gran número de aparatos conectados al tendido. Un simple cortocircuito o un equipo que se calienta demasiado pueden convertirse en la causa directa de un incendio en el edificio, o al menos en la planta, donde se encuentran invertidos millones de pesetas en equipamiento. Un método efectivo contra los incendios son los extintores situados en el techo, que se activan automáticamente al detectar humo o calor.

4.3 Seguridad Jurídica de la estructura de las Bases de Datos

Recientemente han surgido cuestiones acerca de la protección jurídica de las Bases de Datos, así como derechos sobre las mismas que se pueden ostentar. La seguridad jurídica permite un desarrollo en el ámbito de las Bases de Datos, logrando así beneficios a las empresas e incluso a los insumos que contiene una Base de Datos. De hecho, únicamente la protección claramente definida de la

propiedad intelectual puede proporcionar la suficiente seguridad jurídica a los creadores y usuarios, logrando un acceso y uso apropiado de las mismas.

4.3.1 Derechos de autor

Toda persona tiene derecho a formar parte libremente en la cultura de la comunidad, a gozar de las artes y a participar en el progreso científico y en los beneficios que de él resulten. Toda persona tiene derecho a la protección de los intereses morales y patrimoniales que le correspondan por razón de las producciones científicas, literarias o artísticas de que sea autora[2].Para proteger al autor y darle el debido reconocimiento a su obra, o la facultad de oponerse a cualquier modificación sin su consentimiento, así como el uso o explotación por sí mismo o por terceros, existe un conjunto de normas denominado derecho de autor.

Analizando un poco de historia, en 1946, México participó, al igual que 20 países más de América, en la Conferencia Interamericana de Expertos para la Protección de los Derechos de Autor, Unión Panamericana, celebrada en Washington, D.C. En este evento se firmó la Convención Interamericana sobre el Derecho de Autor de obras Literarias, Científicas y Artísticas. Para concordar el Derecho de Autor mexicano, con los compromisos adquiridos en esta Convención, se expidió el 31 de diciembre de 1947, la primera Ley Federal del Derecho de Autor, fue publicada en el Diario Oficial de la Federación el 14 de enero de 1948. Ya en épocas recientes, el 24 de diciembre de 1996, aparece la nueva Ley del Derecho de Autor, que entró en vigor el 24 de marzo de 1997. Ésta nueva Ley da origen al Instituto Nacional del Derecho de Autor, como un organismo desconcentrado de la Secretaría de Educación Pública.

El derecho de autor es el reconocimiento que hace el Estado en favor de todo creador de obras literarias y artísticas.

El derecho de autor es un monopolio legal, de carácter temporal que el Estado otorga a los autores para la explotación de sus obras. Este derecho tiene contenido moral y patrimonial.

4.3.2 Propiedad intelectual

La obra intelectual es la "expresión personal perceptible, original y novedosa de la inteligencia, resultado de la actividad del espíritu, que tenga individualidad, que sea completa y unitaria; que represente o signifique algo, que sea una creación integral susceptible de ser divulgada o reproducida por cualquier medio o procedimiento."

Que se encuentren previstas en el artículo 13 de esta Ley.

[2]Artículo 27 de la Declaración Universal de los Derechos Humanos

Al efecto, el artículo 13 de la Ley dice:

Artículo 13.- Los derechos de autor a que se refiere esta Ley se reconocen respecto de las obras de las siguientes ramas:

I. Literaria, que comprende; libros, folletos y otros escritos;
II. Musical, con o sin letra, con o sin letra;
III. Dramática;
IV. Danza, coreográfica y pantomímica;
V. Pictórica o de dibujo;
VI. Escultórica y de carácter plástico;
VII. Caricaturas e historietas;
VIII. Arquitectónica;
IX. Cinematográfica y demás obras audiovisuales;
X. Programas de radio y televisión;
XI. Programas de cómputo;
XII. Fotográfica; u obra gráfica en serie;
XIII. Obras de arte aplicado que incluyen el De diseño gráfico o textil, y
XIV. De compilación, integrada por las colecciones de obras, tales como las enciclopedias, las antologías, y de obras u otros elementos como las Bases de Datos, siempre que dichas colecciones, por su selección o la disposición de su contenido o materias, constituyan una creación intelectual.

La protección de la propiedad intelectual es un mecanismo para la distribución de contenidos de calidad de acuerdo a términos apropiados y, como tal, ha dado pruebas de los beneficios que podía aportar a la música, la literatura, las películas, los programas informáticos y todo tipo de productos industriales en el entorno tradicional, tanto en el siglo pasado como en la actualidad. Resulta esencial que también las Bases de Datos electrónicas se beneficien de este mecanismo.

Es así como la propiedad intelectual también abarca los programas de computación y las Bases de Datos. La ley federal de derechos de autor define a un programa de computación como la expresión original en cualquier forma, lenguaje o código, de un conjunto de instrucciones que, con una secuencia, estructura y organización determinada, tiene como propósito que una computadora o dispositivo realice una tarea o función específica. En su artículo 102 menciona que:ÒLos programas de computación se protegen en los mismos términos que las obras literarias y que esta protección abarca los programas operativos y los programas aplicativos, ya sea en forma de código fuente o de código objeto. No se protegen los programas de cómputo que tengan por objeto causar efectos nocivos a otros programas o equiposÓ

También menciona en su artículo 103 que: ÑLos derechos patrimoniales sobre un programa de computación y su documentación corresponden a su creador. En caso de que hayan sido creados por uno o varios empleados en el ejercicio de sus funciones o siguiendo instrucciones del jefe, los derechos patrimoniales le corresponderán al jefe, salvo pacto en contrarioÓ

Por su parte, en el artículo 107 señala que:"Las Bases de Datos pueden protegerse como compilaciones siempre que constituyan creaciones intelectuales por razones de selección y por su contenido. Esta protección no comprende datos y materiales en sí mismos".

Sobre los derechos que tiene el autor, el artículo 110 establece:
"El titular del derecho patrimonial de la Base de Datos tendrá el derecho exclusivo, respecto de la forma de expresión de la estructura de dicha base, de autorizar o prohibir:

- Su reproducción permanente o temporal, total o parcial, por cualquier medio y de cualquier forma.
- Su traducción, adaptación, reordenación y cualquier otra modificación.
- La distribución del original o copias de la Base de Datos.
- La comunicación al público, y
- La reproducción, distribución o comunicación pública de los resultados de las operaciones mencionadas en la fracción II del presente artículo".

4.3.3 Protección jurídica de datos

Cabe destacar que la Ley Federal del Derecho de Autor solo protege la estructura de Base de Datos, pero no los datos en sí. El siguiente apartado presenta en qué situación se encuentra la protección de datos a nivel nacional e internacional.

La autodeterminación informativa es un término que define la existencia del derecho del titular sobre los datos que proporciona. De esta forma, quien tenga, mantenga, utilice o trate, en soporte informático, o en forma automatizada o susceptible del tratamiento automatizado, sistemas de datos te carácter personal ï o solo consulta- se encuentra sometido a la normativa conocida como protección de datos que, defiende la llamada privacidad de los ciudadanos, con un órgano de control del cumplimiento de la ley. Este derecho supone que el interesado cuyos datos son objeto de tratamiento puede decidir quién, cuándo y cómo se van a tratar sus datos personales.

La protección se realiza sobre el dato, de manera que éste no pueda ser tratado o elaborado, y convertido en información, nada más que para aquellos fines y por aquellas personas autorizadas para ello. Esta necesaria protección es un límite a la utilización de la Informática ante el temor de que pueda agredir la intimidada de los ciudadanos, personal o familiarmente, y que pueda coartar el ejercicio de derechos.

No se puede tratar ningún dato de carácter personal sin el consentimiento de su titular o interesado. Por tanto se podrán tratar todos aquellos datos para los que el titular de los mismos haya prestado su consentimiento, salvo que concurra alguna excepción prevista en la ley a esta necesidad de consentimiento.

4.3.3.1 Tipo de datos

Los datos personales se conforman con aquella información que hace posible la identificación particular de un individuo. Podemos identificar datos personales íntimos y datos personales de esfera pública. Entre estos últimos se cuentan, principalmente, el nombre y apellidos, fecha de nacimiento, dirección y número de teléfono particulares.

Las diversas legislaciones dan una definición bastante amplia de dato personal. Para las leyes de Austria, Noruega y Dinamarca, es toda información susceptible de ser puesta en relación con personas determinadas o determinables, y se refiere tanto a la persona física como a la personal moral o jurídica en los dos primeros países, mientras la ley danesa refiere sólo a los individuos. La norma de Francia lo define como toda aquella información que permite la identificación directa o indirecta de las personas, mientras la antigua y precursora ley española de 1992 (Ley Orgánica de Tratamiento Automatizado de Datos, LOARTAD) lo define como "cualquier información concerniente a personas físicas identificadas o identificables".

Algunas legislaciones (francesa, sueca, británica) hacen una distinción respecto de los datos que requieren especial protección. Se consideran sensibles los antecedentes policíacos, datos de salud, información de seguridad nacional, creencia religiosa, ideología política, comportamiento sexual e información financiera.

Los datos sensibles, es decir, de la esfera íntima, son aquellos que permiten conocer sobre el patrimonio económico, las enfermedades, las preferencias sexuales, las ideas religiosas, la pertenencia racial o étnica, entre otras que pueden generar prejuicios y discriminaciones que afecten la dignidad, la privacidad, la intimidad, la seguridad, la reputación y la imagen de la persona.

El derecho a la protección de datos personales presenta caracteres propios que le dotan de una naturaleza autónoma de tal forma que su contenido esencial le distingue de otros derechos fundamentales y, específicamente, del derecho a la intimidad, al honor y a la propia imagen.
El derecho a la intimidad tiende a caracterizarse como el derecho a ser dejado solo y a evitar injerencias en la vida privada.

El derecho a la protección de datos atribuye a la persona un poder de disposición y control sobre los datos que le conciernen, partiendo del reconocimiento de que tales datos van a ser objeto de tratamiento por responsables públicos y privados. Dicho tratamiento impone a los responsables una obligación positiva al objeto de que se lleve a cabo con pleno respeto al sistema de garantías propio de este derecho fundamental.

En ocasiones se ha planteado que el derecho a la protección de datos constituye una barrera para la tutela de otros derechos fundamentales o intereses públicos

como la libertad de información, la transparencia y acceso a la información que obre en poder de entidades públicas o el desarrollo de la actividad económica.

Sin embargo, no puede olvidarse que sólo respetando el derecho fundamental de todos a la protección de sus datos personales se conseguirá un marco adecuado de respeto a la libertad de expresión, al acceso a la información y un correcto desarrollo del mercado.

4.3.4 La regulación de la protección de datos en el extranjero

Es necesario partir de los diferentes puntos de vista que se plantean en los ordenamientos jurídicos europeo y de los Estados Unidos.

En el caso de la Unión Europea, tal y como puede comprobarse en la Directiva 95/46/CE y en el Reglamento (CE) n° 45/2001 del Parlamento Europeo y del Consejo, de 18 de diciembre, relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales por las instituciones y los organismos comunitarios y a la libre circulación de esto datos[3],se parte del concepto de sistema de protección de datos señalándose en el considerado 1 del Reglamento (CE) n° 45/2001 que un sistema completo de protección de datos se integra por los siguientes aspectos:

- Derechos de las personas.
- Obligaciones de quienes tratan los datos personales.
- Establecimiento de sanciones apropiadas para los infractores.
- Un organismo de supervisión independiente.

En Europa, la regulación de la protección de datos de carácter personal se caracteriza por su previsión en una norma legal respaldada por la previsión de sanciones para los casos en que se produzca un incumplimiento por quien trata los datos de carácter personal por cualquier título, ya sea un responsable de sistema de datos o un encargado del tratamiento.

En Estados Unidos se basa en la práctica ausencia de previsiones legales, quedando dicha regulación reducida al establecimiento de códigos de conducta o, la denominada, ñautorregulación industrialò desarrollada esencialmente por el sector privado, y cura efectividad depende, en gran parte, del poder de coacción de quien formula dichos códigos y de la aplicación de las sanciones previstas en ellos.

Pero situándonos en México, existe la necesidad de leyes reguladoras. La oportunidad de establecer una regulación específica sobre la protección de datos personales está más que justificada desde diversos puntos de vista, entre los que puede destacarse, en primero lugar, la garantía de un derecho de los ciudadanos como es el derecho a la privacidad y, en segundo lugar, los beneficios que ello reporta tanto al sector privado como al público, en cuanto esta garantía, en caso

[3]Recopilado de la página de http://www.ifai.com.mx

de que cumpla con unos determinados requisitos, determina que se facilite la realización de transacciones comerciales y de otro tipo con la Unión Europea, buscado así además el reconocimiento de un nivel adecuado por parte de la Comisión Europea, lo que supondría la libre realización de transferencias de datos con destino a México.

4.3.5 La protección de datos en México

México se encuentra inmerso en un proceso legislativo de discusión a efecto de generar un marco jurídico aplicable a la protección de datos personales; para ello, el proceso correspondiente debe tomar en consideración la experiencia internacional adquirida en la materia, y permitir un equilibrio eficaz entre el flujo regulado y necesario de la información para promover los mercados en constante desarrollo y expansión, y la protección de la persona en el ámbito de la protección de datos. Adicionalmente, es imperativo que esa ley promueva la cultura de la protección de datos entre los individuos, de forma tal que la misma se convierta en un valor fundamental de nuestra sociedad.

Actualmente no existe una ley específica a nivel federal que regule la protección de datos personales. Aunque las leyes de acceso a la información que se han aprobado en el país en los últimos años contienen capítulos referidos a los datos personales en poder de instancias públicas, lo cierto es que resulta necesaria regular la información personal en poder de entidades o empresas privadas.

En la actualidad el sistema jurídico mexicano cuenta con la Ley Federal de Acceso a la Información Pública Gubernamental, que entró en vigor el 12 de junio del 2002, en donde en un breve apartado regula la protección de los datos personales derivada de los archivos que se manejan para hacerlos públicos a los interesados. También existe la Ley de Información Estadística y Geografía, norma rectora del Instituto Nacional de Estadística Geografía e Informática, publicada en el Diario Oficial de la Federación el 30 de Diciembre de 1980.

Otro ejemplo que se puede mencionar es una Ley que entro en vigor el 15 de enero de 2002 para Regular las Sociedades de Información Crediticia, quienes manejan la información personal de los usuarios de los servicios financieros, en donde se conforman historiales crediticios, y los que deben obedecer a reglas especiales, y de esa manera evitar el uso indiscriminado e injustificado que pudiera darse a la información con que cuentan éstas sociedades en sus Bases de Datos. Cabe agregar que el proyecto inicial disponía las reglas básicas para este tipo de sociedades en materia de datos personales, sin embargo, se suprimió para evitar cualquier conflicto de normas que pudiese suscitarse.

El 30 de abril del 2002, el Senado de la República aprobó una iniciativa del senador del PRI Eduardo García Torres para crear una Ley Federal de Protección de Datos Personales. La ley no ha sido aún promulgada, pues la Cámara de Diputados debe dictaminar sobre esa materia, sobre todo cuando existe un

dictamen sobre la iniciativa presentada el 6 de septiembre de 2001 por el diputado federal del PRD, Luís Miguel Barbosa Huerta, además de que grupos de académicos, así como el senador panista Javier Corral, están trabajando en iniciativas al respecto.

No obstante, existe además el interés de involucrar en esta materia a la Comisión Nacional de Derechos Humanos. Un grupo académico encabezado por el doctor Francisco Javier Acuña e integrado por alumnos del Doctorado en Derecho de la Información que imparten la Universidad de Occidente, la Universidad Iberoamericana y el Instituto de Investigaciones Jurídicas de la UNAM, se ha dado a la tarea de elaborar una iniciativa de Ley Federal de Protección de Datos Personales donde se contempla la intervención reguladora de la CNDH.

La protección de datos personales es una garantía cuya defensa debe recaer en forma natural en la CNDH, dado que se trata de un derecho humano incuestionable. El asunto está en que la institución defensora de derechos civiles no tiene la facultad de sanción, sino solamente la de emitir recomendaciones. Sin embargo, su rango constitucional y su legitimidad lograda convierten sus recomendaciones a una especie de valor casi jurídico.
El panorama en los estados de la república resulta alentador. En Morelos acaba de aprobarse la Ley de Acceso a la Información Pública, Protección de Datos Personales y Estadística, la cual contempla la protección de datos personales en poder de entidades privadas. En Sinaloa, están por presentarse en el Congreso local iniciativas a este respecto, como han anticipado ya el propio Ejecutivo estatal y los diputados del Partido Acción Nacional. También en Quintana Roo tiene en vigor una ley de protección de datos.

Así, una ley de protección de datos personales en México debería considerar, los principios de protección de datos personales internacionalmente reconocidos, aplicables tanto a las entidades públicas como privadas; por otro lado, debe adecuarse a las condiciones sociales, políticas y económicas del país, estableciendo al mismo tiempo reglas claras y sencillas que permitan, no sólo su implementación y cumplimiento, sino también un equilibrio entre el flujo de datos personales y su protección. Por último, la ley debe garantizar el ejercicio efectivo del derecho a la protección de datos de las personas y promover una cultura de confianza y de respeto a los derechos humanos, a través de instituciones independientes.

Conclusiones

El trabajo de investigación desarrollado en esta tesis ha consisitido en exponer los beneficios de contar con un buen diseño de Base de Datos. Para lograrlo se requiere de una planeación previa que puede parecer larga, pero el resultado final simplificará tiempos y costos, dando como consecuencia un uso y explotación óptimo de la información.

Sin embargo, si no se cuenta con un diseño conveniente ni con la documentación necesaria de la Base de Datos con la que se esta trabajando se puede optar por aplicar Reingeniería, mas aún si los requerimientos que se presentan utiliza la misma información contenida en la Base, la solución puede ser el rediseño o reutilización de ésta.

Con el paso del tiempo se busca la modernización y las Bases de Datos deben evolucionar, aunque a veces no es un proceso fácil y además puede ser costoso. De esto se trata la Reingeniería de Base de Datos, tomar la existente y enriquecerla con los requisitos que la empresa necesita. La Ingeniería Inversa forma parte de la Reingeniería y arroja un nivel conceptual de la Base de Datos.

La Ingeniería Inversa, como parte de la reingenieria es una alternativa de renovacion de las Bases de Datos. Es una forma de recuperar el diseño tanto lógico como conceptual y partir de él para optimizar o transformar la Base de Datos de acuerdo a las exigencias que van surgiendo. Además puede proporcionar la documentación que no se tenía y que con ella se podría tener un mayor control de administración sobre la Base y simplificar tareas de proyectos futuros sobre la misma así como mejorar el diseño e implementación de programas que interactuan con la Base de Datos.

Aplicar Ingeniería Inversa significa trabajar con una Base de Datos que es autoria de alguien o alguna organización, esto implica considerar el derecho a la propiedad intelectual. Existe la Ley Federal del Derecho de Autor (LFDA) que protege la estructura y los datos de una Base, sin embargo no hay una ley que mencione expresamente lo relativo a la Ingeniería Inversa sobre Bases de Datos originales y no originales. Sin embargo, la LFDA prevé lo que en un momento puede considerarse como una violación al derecho patrimonial del titular sobre una Base de Datos Por lo cual, para realizar cualquier modificación que se realice a la Base de Datos es pertinente que se cuente con la autorización del titular del derecho patrimonial.

Bibliografía

Adoración de Miguel y Mario Piattini, *Bases de Datos relacionales*. Editorial RA-MA, 1999.

Castaño, Adoración de Miguel, Piattini Velthuis Mario y otros, *Diseño de Bases de Datos Relaciónales.* Editorial RA-MA, 1999.

Davis, William S, *Herramientas Case.* Editorial Paraninfo, 1992.

Eilam Eldad,Elliot Chikofsky, *Reversing Secrets of Reverse Engineering.* Wiley Publishing, 2005.

Luque Ruiz, Irene y Gómez-Nieto, Miguel Ángel, *Bases de Datos, Desde Chen hasta Codd con ORACLE*. Editorial AlfaOmega - RAMA, 1999.

Manganelli, Raymond S. *Como Hacer Reingeniería.* Norma Ediciones S.A. De C. V. ,1997.

Pérez López, Cesar. *Oracle 9i, Administración y Análisis de Bases de Datos.* Editorial AlfaOmega, 2004.

Piattini Velthuis, Mario Gerardo, *Tecnologia Y Diseño De Bases de Datos.* Editorial Alfaomega - RAMA, 2007.

Rozic, Sergio Ezequiel, *Bases de Datos Y Su Aplicación Con Sql*. Mp Ediciones Sa, 2004.

Schiller, Bradley R*,* F*undamentos de Bases de Datos*. Editorial McGraw-Hill, 2001.

Ullman, Jeffrey y Widom, Jennifer, *Introducción a los Sistemas de Bases de Datos*. Editorial Prentice Hall, México 1991.

Fuentes de información

Caja Costarricense de Seguro Social, Dirección de Informática, Departamento de Soporte Técnico. Metodología para el modelo de datos. Publicado en Internet en http://www.ccss.sa.cr/dirinfo/document/mdatinst.pdf. 5 de Noviembre de 2005.

María Mercedes Marqués Andrés. Apuntes de Ficheros y Bases de Datos. http://www3.uji.es/~mmarques/f47/apun/node1.html . 5 de Noviembre de 2005.

http://www.mysql-hispano.org/ Normalización de Bases de Datos. 11 de Marzo de 2006

Alejandro Narancio. Ingeniería Inversa de Base de Datos. http://www.latiumsoftware.com/es/articles/00020.php. 5 de Noviembre de 2005

José L. Leiva Olivencia. Aplicación de Técnicas de Ingeniería Inversa y Reingeniería en Bases de Datos de Sistemas Informáticos de Gestión Hotelera. www.turismo.uma.es/turitec/turitec2004/docs/actas_turitec_pdf/16.pdf . 5 de Noviembre de 2005

Jean HENRARD, Institut d'Informatique - University of Namur Program Understanding in data Reverse Engineering http://www.info.fundp.ac.be/~dbm/publication/2003/jhe_thesis.pdf. 11 de Marzo de 2006

Jean-Luc Hainaut, Laboratory of Database Application Engineering. http://www.info.fundp.ac.be/~dbm/publication/2002/DBRE-2002.pdf. 11 de Marzo de 2006

Secretaria de Educación Pública. ¿Qué es el derecho de autor?. http://www.sep.gob.mx/wb2/sep/sep_1426_que_es_el_derecho_d. 19 de Septiembre de 2006

Secretaria de Educación Pública. Ley federal del derecho de autor. http://www.sep.gob.mx/wb2/sep/sep_Ley_Federal_del_Derecho_de_Autor . 19 de Septiembre de 2006

Instituto Federal de Acceso a la Información Pública**,** Bello, Guzmán, Morales y Tsuru. Estudio sobre Protección de Datos a nivel Internacional. http://www.ifai.org.mx/transparencia/prot_datos.pdf . 19 de Septiembre de 2006

Dorangélica de la Rocha. Protección de datos personales. http://www.mexicanadecomunicacion.com.mx/Tables/RMC/rmc84/proteccion.html. 19 de Septiembre de 2006

Wikipedia. Herramienta CASE. http://es.wikipedia.org/wiki/Herramienta_CASE. 24 de Agosto de 2006