UNIVERSIDAD MAYOR DE SAN ANDRES
FACULTAD DE CIENCIAS PURAS Y NATURALES
CARRERA DE INFORMATICA

TESIS DE GRADO
“MODELO DE OPTIMIZACIÓN PARA EL DISEÑO DE BASES DE DATOS RELACIONALES”

PARA OPTAR AL TITULO DE LICENCIATURA EN INFORMATICA
MENCION : INGENIERIA DE SISTEMAS INFORMATICOS

AUTOR: ALTAMIRANO IBAÑEZ ANDRES MIGUEL

TUTOR: LIC. EFRAÍN SILVA SANCHEZ

REVISOR: LIC. CARLOS MULLISACA CHOQUE

LA PAZ - BOLIVIA

2007

DEDICATORIA

A mis padres Andres y Bruna, por el apoyo en todos los emprendimientos que me propuse y por formarme con los valores éticos y morales necesarios para ser una persona de bien.

A mis hermanas y muy especialmente a mis sobrinos, ya que sin el apoyo moral brindado no se hubiera podido culminar satisfactoriamente este trabajo de investigación.

AGRADECIMIENTOS

Deseo expresar mi eterna gratitud al Lic. Efraín Silva Sanchez y al Lic. Carlos Mullisaca *Choque por el tiempo, los consejos, enseñanzas, paciencia e interés que mostraron para con el presente trabajo.*

Agradecer a la carrera de informática por acogerme todo este tiempo y por enseñarme tanto.

RESUMEN

Cuando se quiere realizar un sistema de información se deben considerar varios aspectos; el presente trabajo se centra en lo que es el almacenamiento de datos, y para este objeto se requiere de una base de datos. Entonces, para tener un buen diseño de las bases de datos se analiza todo lo que ocurre en el entorno del sistema que sea objeto de estudio, desde un punto de vista de la abstracción. Lo primero que se hace es modelar conceptualmente en base a requerimientos de los usuarios, para luego llevarlo a un modelo lógico o comúnmente llamado relacional.

Para un buen diseño del modelo relacional, se deben considera aspectos importantes como lo son la normalización de los datos para evitar la inconsistencia y la redundancia de los datos. Así como la normalización, el principio de diseño ortogonal tiene el objetivo de reducir la redundancia de datos. Por lo que en este trabajo se esta planteando un modelo que resuelva el problema de la existencia de criterios traslapados, mismos que son analizados en el principio antes mencionado.

El modelo que se plantea esta fundamentado de una manera poco ortodoxa como lo es mediante vectores y la definición de proyección ortogonal. Es así que se llega a identificar dos tipos de traslapados de información, como es el traslapado completo y relativo. Las soluciones que da el modelo, en base a un tipo de traslapado esta relacionado con la definición misma del principio de diseño ortogonal. Además, el modelo se basa en la actualización de los datos mediante vistas, mismos que son ingresados en la base de datos.

Para mostrar la aplicabilidad del modelo en el mundo real, se considera mostrar un diseño de una base de datos, que representa a la Gestión de Hospitales, donde casualmente se puede llegar a dar los tipos de traslapados de información antes mencionados. Después de haber aplicado el modelo al diseño que se toma como muestra, se observan cambios, mismos que serán comparados como el antes y el después mediante una métrica para poder así demostrar los beneficios que da el modelo.

UNIVERSITAS MAJOR PACENSIS DIVI ANDREÆ

UNIVERSITAS MAJOR PACENSIS DIVI ANDRE Æ

CONTENIDO

LISTA DE FIGURAS

UNIVERSITAS MAJOR PACENSIS DIVI ANDRE Æ

UNIVERSITAS MAJOR PACENSIS DIVI ANDREÆ

UNIVERSITAS MAJOR PACENSIS DIVI ANDRE Æ

Capitulo 1

INTRODUCCIÓN

1.1INTRODUCCIÓN

En la actualidad por la creciente información que existen en las entidades, tanto públicas y privadas, se tiene la necesidad de contar con sistemas de información, los cuales tienen por finalidad administrar y controlar, valga la redundancia, la información.

Para la elaboración de estos sistemas se suele requerir los servicios de personas entendidas en el área de la informática, los mismos que para el desarrollo deben seguir varias fases, puesto que estos determinaran el éxito o fracaso del desempeño.

Roger S. Pressman en la quinta edición de su libro Ingeniería del Software hace referencia precisamente a los términos éxito y fracaso, por lo que considero importante en este punto introductorio mencionarlo para tenerlo en cuenta.

Cuando un software de computadora se desarrolla con éxito – cuando satisface las necesidades de las personas que lo utilizan; cuando funciona impecablemente durante

mucho tiempo; cuando es fácil de modificar o incluso es más fácil de utilizar – puede cambiar todas las cosas y de hecho las cambia para mejor. Ahora bien, *cuando un software de computadora falla* – cuando los usuarios no quedan satisfechos, cuando es propenso a errores; cuando es difícil de cambiar e incluso más difícil de utilizar – puede ocurrir y de hecho ocurren verdaderos desastres. Todos queremos desarrollar un software que haga bien las cosas, evitando que esas cosas merodeen por las sombras de los esfuerzos fracasados. Para tener éxito al diseñar y construir un software necesitamos disciplina. Es decir, necesitamos un enfoque de ingeniería. ***[PRES03]***

Bien, teniendo en cuenta lo anterior, el tema en sí esta relacionado con lo que es el diseño de la base de datos, el mismo que se debe desarrollar con éxito. Como es de conocimiento de todos los profesionales en informática, cuando se desarrolla una base de datos relacional se la realiza en dos fases. La primera fase esta relacionado con la definición del modelo conceptual y su esquema; la segunda fase trata de transformar el modelo conceptual en un esquema relacional, esto es lo que se realiza frecuentemente.

Siendo mucho más precisos, el objeto de estudio estará orientado en la segunda fase pues, en esta se realiza un proceso de refinamiento denominado normalización. El objetivo de este proceso es de evitar que existan variables relacionales que tengan información redundante e inconsistente.

Lo que se plantea con el presente trabajo es realizar un modelo que sea un complemento al proceso de normalización, dicho modelo estará basado en lo que se denomina principio de diseño ortogonal[1]. Cabe mencionar que para el cumplimiento de la hipótesis y objetivos se considerara en el desarrollo de este trabajo el método inductivo – deductivo, ya que de las observaciones iniciales que se realicen se estipulara una verdad general.

[1] *El término ortogonal se refiere al hecho de que variables relacionales deben ser mutuamente independientes.*

1.2ANTECEDENTES

Los trabajos relacionados que se consultaron y que tienen cierta relación con el presente trabajo son los que se encuentran en la carrera de informática, los cuales se los mencionaran cronológicamente.

Uno de los primeros trabajos relacionados es el de la Lic. Kattia Torrico Saldaña ***[TORR89]*** con el tema "Evaluación del desempeño de Bases de Datos relacional", el cual esta dirigido a una bases de datos relacional en la etapa de producción. Para la evaluación del desempeño toma en cuenta aspectos como: cuando se comienza una evaluación, la administración del sistema en general, la importancia del diseño en el desempeño, el desempeño de la base de datos relacional y por último la administración de la base de datos.

El trabajo realizado por el Lic. Walter Saul Espinoza Torrico ***[ESPI90]*** con el tema "Diseño Automático de Bases de Datos Relacionales", plantea que mediante un conjunto de algoritmos y técnicas aplicadas a unas entradas se puede realizar en forma automática un diseño de una base de datos relacional. Los algoritmos planteados están básicamente para verificar niveles de normalización, los mismos demostrados mediante los axiomas de Amstrong. El alcance del trabajo esta dirigido a bases de datos considerados de mediano tamaño a grande y para diseñadores no muy experimentados en el diseño de base de datos. Para poder demostrar el cumplimiento de los objetivos y la hipótesis, se vio por conveniente presentar un prototipo.

En relación a las consultas de una base de datos, se realizo el trabajo "Optimización de consultas en el diseño de una base de datos relacional", realizado por la Lic. Gladys Francisca Chuquimia Mamani ***[CHUQ98],*** el cual se centra en la etapa del diseño lógico. Ahora bien, este trabajo realiza dicha optimización en base a una definición de entradas, las cuales son representadas por un conjunto de tablas denominadas relación base, los campos de estas tablas se los almacena en una matriz, para aplicar el modelo, que es objeto del desarrollo de la tesis, en términos de operaciones que se realizan con los datos (actualizar, eliminar, etc.) y frecuencia de acceso a cada una de estas operaciones, para luego mediante un cálculo matemático sacar un ponderado de accesos. Finalmente realiza una comparación para hallar las conclusiones pertinentes.

Otro trabajo que se considera importante, al aplicar la inteligencia artificial en el desarrollo de las bases de datos es el denominado "Sistema experto de control de normalización automática de una base de datos relacional", el cual fue desarrollado por el Lic. Constancio Chuquimia Poma ***[CHUQ02]***. Para el cumplimiento de sus objetivos, con relación al tema, se hizo uso de los conceptos de sistemas expertos, el cual por sus características esta dirigido a personas no muy relacionadas con el criterio del buen diseño de una base de datos, más específicamente con el proceso de normalización. Los conocimientos introducidos en el diseño del sistema se los adquirió de algunos docentes de la carrera de informática.

Finalmente para el análisis de la calidad en una bases de datos existe el tema "Evaluación de la calidad de Bases de datos", elaborado por el Lic. Juan Rolando López Maidana ***[LOPE03]***, lo que se plantea en este trabajo es la elaboración de un método basado en métricas[2] que permitan evaluar la calidad del esquema conceptual de una bases de datos relacional. Este trabajo esta dirigido a bases de datos que están listos para ser implementados, esto por razones de que las comparaciones que se hacen están relacionadas con el número de campos, número de llaves candidatas, etc. Uno de los objetivos principales para el desarrollo de este tema es el de entrega de información confiable al ingeniero de software y para facilitar la tarea de una buena auditoria de bases de datos.

Como se puede evidenciar los trabajos mencionados están enfocados al diseño de bases de datos que respondan a cubrir en gran medida las expectativas de los usuarios. Los textos que se mencionan en la bibliografía son un claro ejemplo de la existencia de algunos problemas que todavía se dan, por causa de el crecimiento que se tiene con relación a la información, misma que se hace necesaria para una buena toma de decisiones.

[2] *Cuando podemos medir aquello sobre lo que hablamos, y expresarlo en números, entonces conocemos algo sobre ello; pero cuando no podemos medirlo, ni expresarlo en números, nuestro conocimiento es de tipo pobre e insatisfactorio. (Lord Kelvin)*

1.3DESCRIPCIÓN DEL PROBLEMA

Al realizar el diseño de una base de datos frecuentemente se consideran dos fases, las cuales se pueden describir como sigue:

- *Fase 1:* La cual estaría orientada a definir el modelo conceptual y su esquema.
- *Fase 2:* Está trata de transformar el esquema conceptual en un esquema relacional mediante unas reglas dadas.

En la primera, frecuentemente se hace uso del enfoque Entidad – Relación propuesto por Chen. La base del mismo es considerar la existencia de entidades, las cuales representan personas, objetos, etc., sobre la que se quiere almacenar información relevante. Las características que describen a cada una de las entidades son denominados atributos. Cabe mencionar que el modelo de Chen es n-ario, lo que quiere decir que las relaciones pueden establecerse entre uno, dos o más entidades.

La definición del modelo conceptual según Chen, propone una secuencia de fases para la obtención del modelo:

1. Identificar las entidades dentro del sistema.
2. Determinar las claves o identificadores de entidades.
3. Establecer las relaciones entre las entidades, describiendo el grado de las mismas.
4. Dibujar el modelo de datos.
5. Identificar y describir los atributos de cada entidad.
6. Verificaciones.

El modelo obtenido es una representación gráfica especializada.

La segunda fase trata de realizar un refinamiento al modelo conceptual convirtiéndolo en un modelo lógico relacional, aplicando los criterios que plantea la normalización, lo que da como resultado el conjunto de tablas a ser implementadas en la base de datos.

El procedimiento de normalización consiste en someter a las tablas, que representan entidades, a un análisis formal para ver si cumplen o no las restricciones necesarias que aseguren evitar redundancia de datos, evitando así las anomalías de actualización. A mayor grado de normalización mejor diseño de la base de datos. Se considera la existencia de cinco formas normales, pero lo más deseable es la forma normal de Boyce – Codd[3] (FNBC), que es considerada una mejora a la tercera forma normal (FN3), la misma que fue planteada por Boyce.

Ahora bien, a partir de todo este análisis realizado, se puede mencionar algunos problemas que se encuentran también en el diseño:

- La mala interpretación semántica por parte del diseñador de la base de datos
- Una falta o mala interpretación de las restricciones, lo que implica tener un mal diseño del diagrama relacional.
- Cuando se define un diagrama Entidad – Relación, las tablas generadas pueden no necesitar más normalización, pero pueden haber dependencias funcionales entre los atributos de una entidad.
- La existencia de una tupla o parte de ella en dos o más variables relacionales, lo que implica tener redundancia.
- La mala normalización de los esquemas relacionales.
- No contar con la información actualizada cada vez que se hace uso de una aplicación.

Ante la existencia de estos problemas se construye la pregunta de investigación:

¿El establecimiento de un modelo para optimizar el diseño lógico de las bases de datos solucionará el problema de las variables relacionales con significados traslapados?

[3] *En 1970 el Dr. Edgar F. Codd de IBM propuso un modelo relacional generalizado, orientado a conseguir independencia de datos.*

1.4HIPÓTESIS

Para el planteamiento de la hipótesis del presente trabajo se ha considerado el de tipo causal, ya que esta establece relaciones de causa – efecto. A las variables "causa" se le denomina variable independiente y a los efectos o consecuencias, variables dependientes ***[SARD04]***. Entonces la formulación es la siguiente:

Def.

"El modelo de optimización en el diseño de bases de datos relacionales evita que dos o más variables relacionales tengan significados traslapados".

Operacionalización de la hipótesis

Variable Independiente: Incremento de variables relacionales que tienen significados que se traslapan.

Variables Dependiente: Existencia de redundancia de datos.

Hipótesis	Variables	Indicadores
El modelo de optimización en el diseño de bases de datos relacionales evita que dos o más variables relacionales tengan significados traslapados.	**Unidad de análisis:** Diseño de bases de datos relacionales **Variables:** **1.** Incremento de variables relacionales que tienen significados que se traslapan. **2.** Existencia de redundancia de datos.	**1.1.** Existencia de variables relacionales con encabezados iguales o similares. **1.2.** Un mal análisis del cálculo de dependencias. **1.3.** La mala interpretación de requerimientos de los usuarios. **1.4.** Mala normalización de los esquemas que conforman el sistema. **2.1** Mayor uso de recursos tecnológicos. **2.2** Presencia de anomalías de actualización de los datos. **2.3** Los datos son más complicados, en términos del entendimiento de los mismos.

Hipótesis	Variables	Definición
El modelo de optimización en el diseño de bases de datos relacionales evita que dos o más variables relacionales tengan significados traslapados.	**Unidad de análisis:** Diseño de bases de datos relacionales **Variables:** **1.** Incremento de variables relacionales que tienen significados que se traslapan. **2.** Existencia de redundancia de datos.	Se entiende por variables relacionales con significados traslapados a aquellas tuplas que pueden llegar a satisfacer los predicados de dos o más variables relacionales, mismas que serán todas las expresables. Se entiende por redundancia de datos cuando una misma información esta duplicada en diferentes lugares (archivos).

1.5OBJETIVOS

1.5.1 Objetivo General

Establecer un modelo formal que optimice el diseño lógico de las bases de datos, con el cual se solucionara el problema de que existan variables relacionales con significados traslapados[4].

1.5.2 Objetivos Específicos

a. Plantear un modelo que sea considerada una herramienta más para el diseñador de bases de datos, esto para reducir el tiempo de trabajo que pueda implicar errores que se presenten a futuro.
b. Analizar las formas normales existentes para comprobar que ninguna aborda el problema planteado.
c. Determinar de que manera coadyuvara para el presente trabajo el tema de vistas, tabulación cruzada y el principio de diseño ortogonal.

[4] *Son significados traslapados cuando una misma tupla satisface los predicados de dos o más variables relacionales.*

d. Verificar el modelo con un prototipo, mismo que será debidamente normalizado para aplicar luego el modelo planteado.
e. Incorporar el modelo que optimice el diseño de bases de datos en la carrera de informática para su divulgación y cumplimiento.

1.6JUSTIFICACIÓN

1.6.1 Justificación técnico – científica

El presente trabajo se justifica científicamente, ya que se considera en la elaboración del modelo el principio de diseño ortogonal, en vista de que este no es parte del proceso de normalización en sí, pero si se asemeja a ella debido a que es científico.

El objetivo de este principio de diseño es el de reducir la redundancia y evitar así las anomalías de actualización. Para ser más específicos complementa la normalización, en el sentido de que – en general – la normalización reduce la redundancia dentro de las variables relacionales, mientras que la ortogonalidad la reduce a través de las variables relacionales. ***[DATE01]***

1.6.2 Justificación Social

Al considerar el diseño de bases de datos relacionales, el presente modelo será una herramienta más para el diseñador del mismo, siendo este un complemento para evitar la redundancia de datos, en un sentido orientado a dar solución de problemas con significados de variables relacionales que se traslapan. Se espera ser un aporte importante en el diseño de las bases de datos.

1.7METODOLOGÍA

El método que será considerado para el desarrollo de esta tesis de grado será el método inductivo – deductivo, por las características que este presenta. Para poder entender mejor este método, considero importante explicar los métodos inductivo y deductivo respectivamente para tener claro el porque de la elección del método.

1.7.1 Método inductivo

"Es el proceso de razonamiento de una parte de un todo; va de lo particular a lo general, de lo individual a lo universal". ***[ROSA90]***

El método inductivo utiliza la información generada por otros métodos, tales como: el de casos y el estadístico, para tratar de inducir una relación que incluya no sólo los casos particulares, sino que permita su generalización. Es decir, el método inductivo se apoya en los resultados de algunos casos particulares para establecer una relación general, por ejemplo, se hace una inferencia inductiva cuando decimos: "Todos los mamíferos son seres vertebrados"; tomando como base nada más la observación de algunos de ellos.

Una de las tareas más significativas del intelecto, es justamente la aplicación de este método, ya que no cuenta con reglas para su procedimiento. Así pues, podemos concluir que el método inductivo es un procedimiento de sistematización en el que, a partir de resultados particulares se van a buscar las relaciones generales que las expliquen. ***[DEGO79]***

1.7.2 Método deductivo

Es el procedimiento de razonamiento que va de lo general a lo particular, de lo universal a lo individual. Es importante señalar que las conclusiones de la deducción son verdaderas, si las premisas de las que parte también lo son. A diferencia del método inductivo, en donde la forma de inferencia es de tipo deductivo, al respecto Rosas y Riveros señalan que para Creigthon y Smart, [...] "la inducción y la deducción son formas de inferencia y [que] es un error considerarlas como dos formas de razonamiento diferentes", ya que ambas alcanzan el mismo propósito, pero desde un punto de partida distinto. ***[ROSA90]***

1.7.3 Método inductivo - deductivo

La información ofrece muchas informaciones que el científico percibe, a primera vista, como datos desorganizados. Al manipular esos datos, mediante un proceso que se llama inducción, elabora una hipótesis. Las hipótesis permiten, mediante un proceso llamado

deducción, organizar los datos en forma de leyes, teorías y modelos. Las leyes, teorías y modelos deben ser contrastados con la realidad reanudándose así el proceso. ***[COLL02]*** Entonces el proceso que se sigue es:

- Observación.
- Probar que la observación es confiable.
- Se genera conocimiento científico por inducción.

1.8 ALCANCES Y APORTES

El presente trabajo de tesis se limitará a los siguientes aspectos:

- Se tomará como base el modelo relacional.
- El modelo planteado estará completamente inmiscuido con los conceptos que se dan para la construcción del modelo conceptual, los mismos que coadyuvaran para el desarrollo del presente trabajo.
- Se abordara el estudio a detalle de lo que es el principio de diseño ortogonal.
- El trabajo abarcara conceptos que relacionan las vistas materializadas, tabulación cruzada y el cálculo de dependencias.

En cuanto a los aportes presentados por este trabajo son:

- El aporte del modelo que se presenta es el de dar una solución a la repetición de algunas variables relacionales; por lo que más que ser otra forma normal, lo que se quiere que sea es una herramienta para el diseñador de bases de datos.
- De esta manera este trabajo proporciona aportes al área del diseño y administración de base de datos.

Capitulo 2

MARCO TEÓRICO

2.1 INTRODUCCIÓN

Es importante considerar, como punto introductorio, que para tener un buen diseño de las bases de datos, se debe analizar todo lo que ocurre en el entorno del sistema[5] que se esta analizando, desde un punto de vista de la abstracción[6]. Por lo que el presente trabajo abordara todo lo que este relacionado al diseño de una base de datos, centrándose, como objetivo principal, en lo que es el modelo de datos relacionales.

Entender lo que es un modelo es de vital importancia, ya que en el transcurso del trabajo se mencionara bastante, mas adelante en un punto específico se entrara en detalle con respecto a este término, ahora nos conformamos con la definición que nos da Flory (1982), "modelar consiste en definir un mundo abstracto y teórico, talque las conclusiones que se puedan sacar de él coincidan con las manifestaciones aparentes del mundo real".

---

[5] *Consideraremos lo que dice la Real Academia de la Lengua, en relación a sistema: "conjunto de cosas que ordenadamente relacionadas entre sí, contribuyen a un determinado objetivo". El objetivo que persigue el sistema es común al de sus componentes.*

[6] *La abstracción nos ayuda para modelar los datos al hacer que nos centremos en lo esencial, pasando por alto aspectos que no se consideran relevantes para los objetivos en la representación del mundo real.*

Ahora bien, sabiendo lo que es un modelo, claro someramente, estamos en condiciones de abordar con más propiedad los modelos que sirven para el diseño de las bases de datos. El punto de partida será el modelo Entidad - Interrelación[7] (E - R), propuesto por Peter P. Chen, en sus dos artículos considerados históricos, pero que aun tienen peso en el campo que se esta estudiando.

Según Chen (1976): "El modelo E – R puede ser usado como una base para una vista unificada de los datos", adoptando "el enfoque más natural del mundo real que consiste en entidades e interrelaciones" ***[ADOR00]***

Chen con el afán de no dejar dudas sobre lo que afirma, tienen por bien explicar lo que se entiende por Entidad como "una cosa que se puede identificar claramente", esto puede ser objetos, personas, etc., y por Interrelación como una "vinculación". Este modelo es considerado n-ario, lo que quiere decir que las interrelaciones pueden establecerse entre una, dos o más entidades. También es llamado modelo conceptual, en vista que permite concebir la base de datos a un nivel superior de abstracción, aislándolo de consideraciones relativas a la máquina (tanto en su nivel lógico como físico) y a los usuarios en particular (nivel externo), y centrándolo en un plano de información lógica en el que la información desempeña un papel fundamental.

Para la definición del modelo conceptual con la técnica propuesta por Chen propone una secuencia de fases para la obtención del modelo, y son: Identificar las entidades dentro del sistema, determinar las claves o identificadores de entidades, establecer las interrelaciones entre las entidades, dibujar el modelo de datos, identificar y describir los atributos de cada entidad, y por último realizar las verificaciones.

El tema en sí por el que se esta realizando este trabajo de investigación esta más relacionado con lo que se a denominado modelo de datos relacional, planteado en sus inicios por el Dr. E. F. Codd en el año 1970, en su celebre articulo presentado a la ACM, titulado: "Un modelo de datos relacional para grandes bancos de datos compartidos", mismo que se constituyo en un hito en la historia de las bases de datos.

---

[7] *Se denominara modelo Entidad - Interrelación para no tener confusiones al hacer referencia a una relación, ya que esta es considerada una tabla.*

El objetivo principal que se persigue con el modelo de datos relacionales, es el de aislar al usuario de las estructuras físicas de los datos, consiguiendo así la independencia de las aplicaciones respecto a los datos, finalidad que es perseguida desde los inicios de las bases de datos.

Antes de la aparición del modelo de datos relacional, se organizaban los datos en forma de listas compartidas, jerárquicos y de red, mismos que se manejaban como listas. Las ventajas que se tiene haciendo uso del modelo de datos relacionales a comparación de las antes mencionadas son: la sencillez y uniformidad, la sólida fundamentación teórica y por último la independencia de la interfaz del usuario.

Hasta aquí, como se habrá notado, se ha dado un vistazo a grandes rasgos de lo que se hizo en la historia, con relación al diseño de bases de datos. Ahora en el transcurso de este capitulo se aclararan muchos conceptos, basados en la teoría matemática.

2.2 DEFINICIÓN DE MODELO

Cuando se realiza la definición de requisitos y del diseño conceptual se identifican las exigencias de los usuarios para representar estos en un modelo bien definido. Entonces, *¿Qué es un modelo?* Un modelo es una representación de la realidad que conserva sólo los detalles relevantes. ***[HANS00]***

Se puede dar como ejemplo de un modelo a una base de datos, mismo que incorpora un modelo de la realidad. El sistema gestor de bases de datos (SGBD[8]) administra la base de datos de modo que cada usuario pueda registrar, acceder y manipular los datos que constituyen su modelo de la realidad. Manipulando los datos en una amplia variedad de formas, los usuarios pueden obtener la información necesaria para conducir una empresa con éxito. Por lo tanto, los modelos son herramientas poderosas para eliminar los detalles irrelevantes y comprender la realidad de los usuarios individuales.

Modelar la realidad es en muchas maneras similar a resolver un problema de una historieta. Ambos requieren que nos desprendamos de los detalles para crear un modelo "correcto" de una porción de la realidad. Esto significa que se debe asociar, o identificar,

[8] *SGBD consiste en una colección de datos interrelacionados y una colección de programas para acceder a esos datos. El objetivo principal es el de proporcionar un entorno que sea tanto conveniente como eficiente para las personas que lo usan para la recuperación y almacenamiento de la información.*

elementos de la realidad con elementos en el modelo. Si esta asociación se hace correctamente, entonces el modelo se puede usar para resolver el problema. De lo contrario, el modelo no puede producir una solución correcta.

Bajo la estructura de la base de datos se encuentra el modelo de datos[9]. Para ilustrar el concepto de un modelo de datos, se describen dos modelos de datos: el modelo entidad relación y el modelo relacional.

2.3 EL MODELO RELACIONAL

Existen; hoy en día; diversos modelos para el diseño de bases de datos; así también cada uno utiliza alguna herramienta matemática para la descripción de la base de datos, por ejemplo: en los Modelos de Red se usan herramientas Semánticas, en los Orientados a Objetos se utilizan los grafos; en el Modelo Jerárquico, los árboles; y en el Modelo Relacional, las relaciones.

El *Modelo Relacional* fue presentado por Codd en una publicación en 1970, refiriéndose a un modelo de datos en específico, que se caracterizaba por contar con *Relaciones* como único objeto de tratamiento en el modelo, así también definió un *álgebra* como lenguaje de consulta a la que llamó *Álgebra Relacional* (AR), pero no contaba con ninguna manera de expresar actualizaciones, restricciones y/o cálculos sobre el modelo. ***[FLOR06]***

En el *Modelo Relacional* se utiliza un grupo de tablas para representar los datos y las relaciones entre ellos. Cada tabla esta compuesta por varias columnas, y cada columna tiene un nombre único. En la Figura 2.1 se presenta un ejemplo de base de datos relacional consistente en tres tablas: la primera muestra los clientes de un banco, la segunda las cuentas, y la tercera, las cuentas que pertenecen a cada cliente.

[9] *Es una colección de herramientas conceptuales para describir los datos, las relaciones, la semántica y las restricciones de consistencia.*

Id-cliente	*Nombre-cliente*	*Calle-cliente*	*Ciudad-cliente*
19.283.746	Gonzáles	Arenal	La Granja
01.928.374	Gómez	Carretas	Cerceda
67.789.901	López	Mayor	Peguerinos
18.273.609	Abril	Preciados	Valsaín
32.112.312	Santos	Mayor	Peguerinos
33.666.999	Rupérez	Rambias	León
01.928.374	Gómez	Carretas	Cerceda

(a) La tabla *cliente*

Número-cuenta	*Saldo*
C-101	500
C-215	700
C-102	400
C-305	350
C-201	900
C-217	750
C-222	700

(b) La tabla *cuenta*

Id-cliente	*Número-cuenta*
19.283.746	C-101
19.283.746	C-201
01.928.374	C-215
67.789.901	C-102
18.273.609	C-305
32.112.312	C-217
33.666.999	C-222
01.928.374	C-201

(c) La tabla *impositor*

FIGURA 2.1 Ejemplo de base de datos relacional

El *modelo relacional* es un ejemplo de un modelo basado en registros[10]. Los modelos basados en registros se denominan así porque la base de datos se estructura en registros de formato fijo de varios tipos. Cada tabla contiene registros de un tipo particular. Cada tipo de registro define un número fijo de campos, o atributos. Las columnas de la tabla corresponden a los atributos del tipo de registro.

El *modelo relacional* es el modelo de datos más ampliamente usado, y una amplia mayoría de sistemas de bases de datos actuales se basan en el modelo relacional. Este modelo se encuentra a un nivel de abstracción inferior al modelo de datos E-R. ***[KORT02]***

[10] *Los registros son un conjunto de filas. Matemáticamente será tupla. Por ejemplo, en la tabla cuenta se tiene: C-101 que es el número de cuenta, y este tiene un saldo de 500, en este caso toda esta información pertenece a un registro.*

El *modelo relacional* se propone, como principal objetivo, aislar al usuario de las estructuras físicas de los datos, consiguiendo así la independencia de las aplicaciones respecto de los datos, finalidad perseguida desde los inicios de las bases de datos.

En el nuevo modelo, basado en la teoría matemática de las relaciones, los datos se estructuran lógicamente en forma de relaciones[11]. Esta formalización matemática convirtió rápidamente al modelo en una fuente fundamental de la investigación en bases de datos. ***[ADOR00]***

2.3.1 Dominios, Atributos, Relaciones y Tuplas

Para ser mucho más específicos, en cuanto a estos cuatro términos se ha visto por conveniente dar una definición general en principio, para luego formalizar matemáticamente.

2.3.1.1 Dominio

El Universo del Discurso (UD) de una base de datos relacional esta compuesto por un conjunto de dominios $\{D_i\}$ y de relaciones $\{R_i\}$ definida sobre los dominios.

Un dominio no es mas que un tipo de datos (para abreviar, un *tipo*); posiblemente un tipo simple definido por el sistema como INTEGER o CHAR o, en forma más general, de hecho, podemos usar los términos *tipo*[12] *y dominio* de manera indistinta. ***[DATE00]***

Entonces, más formalmente, un dominio es un conjunto nominado, finito y homogéneo[13] de valores atómicos. Cada dominio se especifica lógicamente mediante un nombre y un formato, el cual puede definirse por extensión (dando sus posibles valores) o por intención (mediante un tipo de datos). A veces se asocia al dominio su unidad de medida (Kilos, metros, etc.) y ciertas restricciones (como un rango de valores).

---

[11] *El término relación viene a ser considerado como sinónimo de tabla, aunque más adelante se refutara esta afirmación. Se usa el término tabla en lugar de relación a fin de simplificar la nomenclatura de cara al usuario final.*

[12] *Cuando se hable de tipo se considerara como un conjunto de valores, todos los valores posibles (ej. Integer es un conjunto de todos los enteros posibles).*

[13] *Son valores homogéneos porque todos son del mismo tipo.*

2.3.1.2 Atributo

Las entidades[14] se describen en una base de datos mediante un conjunto de atributos. Por ejemplo, los atributos *Número-cuenta* y *Saldo* describen una cuenta particular de un banco y pueden ser atributos del conjunto de entidades *cuenta*. ***[KORT02]***

Un atributo (A) es la interpretación de un determinado dominio en una relación, es decir el "papel" que desempeña en la misma; si D es el dominio de A se denota:

$$D = Dom\ (A)$$

Por ejemplo, en una relación CURSO, un atributo puede ser *Cód_curso* y otro *Matería* definidos, respectivamente, sobre los dominios: *Códigos* y *Materias*.

Un atributo y un dominio pueden llamarse igual, pero hay que tener en cuenta que:

- Un atributo esta siempre asociado a una relación, mientras que un dominio tiene existencia propia por independencia de las relaciones.
- Un atributo representa una propiedad de una relación.
- Un atributo toma valores de un dominio.
- Varios atributos distintos (de la misma o de diferentes relaciones) pueden tomar sus valores del mismo dominio.

Matemáticamente, una relación definida sobre un conjunto de dominios $D_1 \ldots D_n$ (no necesariamente distintos) es un subconjunto del producto cartesiano de los *n* dominios (*n* es el grado de la relación). ***[ADOR00]***

2.3.1.3 Relación

Una relación es una asociación entre varias entidades. Por ejemplo, una relación *Impositor* asocia un cliente con cada cuenta que tiene. ***[KORT02]***

[14] *Una entidad es una "cosa" u "objeto" en el mundo real que es distinguible de otros objetos.*

El término relación es básicamente el término matemático para *tabla*; de hecho, los términos *relación* y *tabla* pueden tomarse como sinónimos, por lo menos para fines informales. ***[DATE01]***

Podemos precisar mejor el concepto de relación si lo definimos en base a sus atributos distinguiendo entre el esquema de relación y relación; un ***esquema[15] de relación*** se compone de un nombre de relación R, de un conjunto de n atributos $\{A_i\}$ y de un conjunto de n dominios (no necesariamente distintos) $\{D_i\}$, donde cada atributo será definido sobre un dominio:

$$R\ (A_1 : D_1, A_2 : D_2 ..... A_n : D_n)$$

Un ejemplo de un esquema de relación es el siguiente:

Esque-empresito = (nombre-sucursal, ciudad-sucursal, activo, nombre-cliente, número-préstamo, importe)

nombre-sucursal	*ciudad-sucursal*	*activo*	*nombre-cliente*	*número-préstamo*	*importe*
Centro	Arganzuela	9.000.000	Santos	P-17	1.000
Moralzarzal	La Granja	2.100.000	Gómez	P-23	2.000
Navacerrada	Aluche	1.700.000	López	P-15	1.500
Centro	Arganzuela	9.000.000	Sotoca	P-14	1.500
Becerril	Aluche	400.000	Santos	P-93	500
Collado Mediano	Aluche	8.000.000	Abril	P-11	900
Navas de la Asunción	Alcalá de Henares	300.000	Valdivieso	P-29	1.200
Segovia	Cerceda	3.700.000	López	P-16	1.300
Centro	Arganzuela	9.000.000	Gonzáles	P-18	2.000
Navacerrada	Aluche	1.700.000	Rodríguez	P-25	2.500
Galapagar	Arganzuela	7.100.000	Amo	P-10	2.200

FIGURA 2.2 Ejemplo de base de datos relacional

Una relación se representa utilizando una tabla donde:

- Las columnas de la tabla son los atributos que expresan las propiedades de la relación. El número de atributos se llama **grado** de la relación.

[15] *J. Martin (1977) considera fundamental la elaboración de un esquema como parte del proceso de diseñar una base de datos. El esquema constituye una descripción lógica de la base de datos, materializada en un diagrama que muestra los nombres de las entidades y sus atributos -esto es, los tipos de registro- y las vinculaciones entre ellos. El esquema muestra todas las entidades y sus relaciones.*

- Cada fila de la tabla, llamada tupla, es un elemento del conjunto que es la relación. El número de tuplas se llama **cardinalidad** de la relación. La cardinalidad varía en el transcurso del tiempo.

No se deben confundir los conceptos de tabla y de relación, puesto que:

- Una tabla es una forma de representar una relación.
- Una relación tiene unas propiedades intrínsecas que no tienen una tabla, y que se derivan de la misma definición matemática de relación, ya que, al tratarse de un conjunto, en una relación:

    - No puede haber dos tuplas iguales.
    - El orden de las tuplas no es significativo.
    - Cada atributo sólo puede tomar un único valor del dominio simple subyacente; no se admiten grupos repetidos (ni otro tipo de estructuras) como valores de los atributos de una tupla. ***[ADOR00]***

Una relación de grado 1, se llama *unaria*; una de grado 2, *binaria*; una de grado 3, *ternaria*, y una de grado *n*, *n – aria*. Las características de una relación son:

i. Todas las entradas o elementos de una columna son del mismo tipo.
ii. Se asignan nombres distintos a las columnas, llamados nombres de atributo.
iii. El orden de las columnas es indiferente.
iv. Cada tupla es distinta, esto es, no se permiten tuplas duplicadas.
v. El orden de las tuplas es indiferente. ***[DEEN87]***

2.3.1.4 Tupla

Cada columna de una relación contiene los valores de un atributo con nombre, y cada fila se llama *tupla*[16]. La palabra *tupla* se toma de la definición de grupos tales como quíntupla, séxtupla, etc. Así pues, un grupo de *n* elementos es una *n – tupla*. En una relación de *n* columnas, cada tupla, esto es cada fila, es una *n – tupla*. El número de filas o tuplas de una relación es su ***cardinalidad***, y el número de columnas es su ***grado***. Los elementos individuales de una relación son valores de atributo. Si consideramos una relación *m x n* (*m* filas y *n* columnas), tendremos:

> *Una relación de grado n y cardinalidad m, o sea: una relación que contiene n columnas y m tuplas, siendo cada tupla una n – tupla. Hay m x n valores de atributo, teniendo cada tupla n valores de atributo.* ***[DEEN87]***

En términos generales una tupla corresponde aproximadamente a la noción de una fila (tal como el término relación corresponde aproximadamente a la noción de una tabla). Las tuplas están en desorden, de arriba hacia abajo, esa propiedad también surge del hecho de que el cuerpo de la relación es un conjunto matemático[17]. En otras palabras, no existe el concepto de direccionamiento posicional y no existe el concepto de "la que sigue".

Una relación r(R) es un conjunto de m elementos denominados tuplas $\{t_j\}$. Cada tupla es un conjunto de pares $(<A_1 : v_{1j}>,....<A_i : v_{ij}>,....<A_n : v_{nj}>)$ donde cada A_i es el nombre de un atributo y v_{ij} es un valor del correspondiente dominio D_i sobre el que esta definido el atributo: ***[ADOR00]***

$$r(R) = t_j\{(<A_1 : v_{1j}>,....<A_i : v_{ij}>,....<A_n : v_{nj}>) : v_{ij} \in D_i\}$$

Gráficamente se puede visualizar en la figura 2.3:

---

[16] *Con relación al término tupla (haciendo una comparación entre tabla y relación), recordemos que en una tabla puede contener filas duplicadas - en esencia de cualquier disciplina que evite tal cosa -, mientras que una relación no puede contener una tupla duplicada. Otra diferencia es que las filas de una tabla tienen un orden de arriba hacia abajo, mientras que las tuplas de una relación no las tienen.*

[17] *En matemáticas, los conjuntos no están ordenados.*

EMP

EMP# A_1	EMP# v_{ij}	NOMEMP	NOMBRE	DEPTO#	DEPTO#	SALARIO	DINERO
E1		López		D1		40K	
E2		Cheng		D1		42K	
E3		Pérez		D2		30K	
E4		Hernández		D2		35K	

FIGURA 2.3 Ejemplo del valor relación EMP, que muestra los tipos de columnas

2.3.2 Conjunto de entidades

Una *entidad* es una "cosa" u "objeto" en el mundo real que es distinguible de todos los demás objetos. Por ejemplo cada persona en un desarrollo es una entidad. Una entidad tiene un conjunto de propiedades, y los valores para algún conjunto de propiedades pueden identificar una entidad de forma unívoca.

Un *conjunto de entidades* es un conjunto de entidades del mismo tipo que comparten las mismas propiedades, o atributos. El conjunto de todas las personas que son clientes en un banco dado, por ejemplo, se pueden definir como el conjunto de entidades *cliente*. Análogamente, el conjunto de entidades *préstamo* podría representar el conjunto de todos los préstamos concedidos por un banco particular. Las entidades individuales que constituyen un conjunto se llaman la *extensión* del conjunto de entidades. Así todos los clientes de un banco son la extensión del conjunto de entidades *cliente*.

Una entidad se representa mediante un conjunto de atributos. Los atributos describen propiedades que posee cada miembro de un conjunto de entidades. La designación de un atributo para un conjunto de entidades expresa que la base de datos almacena información similar concerniente a cada entidad del conjunto de entidades; sin embargo, cada entidad puede tener su propio valor para cada atributo.

Cada entidad tiene un valor para cada uno de sus atributos. Por ejemplo, una entidad cliente en concreto puede tener el valor 32.112.312 para *id-cliente*, el valor Santos para *nombre-cliente*, el valor Mayor para *calle-cliente* y el valor Pegueritos para ciudad-cliente.

2.3.3 Conjunto de relaciones

Una *relación* es una asociación entre diferentes entidades.

Un *conjunto de relaciones* es un conjunto de relaciones del mismo tipo. Formalmente es una relación matemática con n >= 2 de conjunto de entidades (posiblemente no distintos). Si $E_1, E_2, \ldots, E_n$ son conjunto de entidades, entonces un conjunto de relaciones *R* es un subconjunto de:

$$\{(e_1, e_2, \ldots, e_n) / e_1 \in E_1, \ e_2 \in E_2, \ldots, \ e_n \in E_n\}$$

Donde $(e_1, e_2, \ldots, e_n)$ es una relación. ***[KORTH02]***

2.3.4 Variables relacionales

Para poden entender de mejor manera, observe la figura 2.4:

EMP

EMP#	NOMEMP	DEPTO#	SALARIO
E1	López	D1	40K
E2	Cheng	D1	42K
E3	Pérez	D2	30K
E4	Hernández	D2	35K

EMP'

EMP#	NOMEMP	DEPTO#	SALARIO
E1	López	D1	40K
E2	Cheng	D1	42K
E3	Pérez	D2	30K

FIGURA 2.4 La variable de relación *EMP*, y *EMP'* después de liminar la fila E4

De manera conceptual, lo que sucedió aquí es que el valor de relación de EMP fue reemplazado en bloque por un *valor de relación completamente nuevo* (EMP'). Desde luego, el valor anterior (EMP con cuatro filas) y el nuevo (EMP' con tres filas) son muy similares, pero de manera conceptual son valores *diferentes*. De hecho la operación de eliminación en cuestión es básicamente una forma abreviada de una cierta operación de **asignación relacional** que podría ser como la siguiente:

EMP' : = EMP MINUS (EMP WHERE EMP# = 'E4');

Desde luego, las operaciones relacionales INSERT y UPDATE también son en esencia formas abreviadas de ciertas asignaciones relacionales.

Ahora bien, desgraciadamente gran parte de la literatura usa el término *relación* cuando en realidad se refiere a una *variable relacional* (al igual que cuando se refiere a una relación como tal; es decir, un valor de relación). Sin embargo, esta práctica ha llevado ciertamente a una confusión. Por lo que en adelante se usara el término VARREL como una abreviatura conveniente de variable relación (o variable relacional) y se tendrá especial cuidado de enunciar las observaciones en términos de varrels (y no de relaciones) cuando realmente se refiere a las variables relacionales. ***[DATE01]***

2.3.5 Claves

Es necesario tener una forma de especificar cómo las entidades dentro de un conjunto de entidades dado y las relaciones dentro de un conjunto de relaciones dado son distinguibles. Conceptualmente las entidades y relaciones individuales son distintas; desde una perspectiva de bases de datos, sin embargo, la diferencia entre ellas se debe expresar en términos de sus atributos.

Por lo tanto, los valores de los atributos de una entidad deben ser tales que permitan *identificar unívocamente*[18] a la entidad. En otras palabras, no se permite que ningún par de entidades tengan exactamente los mismos valores de sus atributos. ***[KORT02]***

2.3.5.1 Superclave

Una superclave es un conjunto de uno o más atributos que, tomados colectivamente, permiten identificar de forma única una entidad en el conjunto de entidades. Por ejemplo, el atributo *id-cliente* del conjunto de entidades *cliente* es suficiente para distinguir una entidad cliente de las otras. Así, *id-cliente* es una superclave. Análogamente es una superclave del conjunto de entidades *cliente*. El atributo *nombre-cliente* de cliente no es una superclave, porque varias personas podrian tener el mismo nombre.

[18] *Término que se refiere a la propiedad de unicidad, lo que significa que ningún par de tuplas que esté dentro del valor de una relación en un momento dado, pueden estar duplicadas entre sí.*

El concepto de superclave no es suficiente para lo que se propone, ya que, como se ha visto, una superclave puede contener atributos innecesarios. Si *K* es una superclave, entonces también lo es cualquier superclave de *K*. A menudo interesan las superclaves tales que los subconjuntos propios de ellas no son superclaves. Tales superclaves mínimas se llaman *claves candidatas*. ***[KORT02]***

2.3.5.2 Clave candidata

Una clave candidata de una relación es un conjunto no vacío de atributos (descriptor) que identifican unívocamente y minimamente cada tupla de una relación.

En toda relación siempre hay, al menos, una clave candidata, ya que el conjunto de atributos que constituye la relación gozará de la propiedad de unicidad (por la definición misma de relación no puede haber dos tuplas iguales) y, si no tuviese la de minimalidad, se podría prescindir de aquellos atributos que lo impidieran, obteniendo así una clave candidata.

Una relación puede tener más de una clave candidata, entre ellas cabe distinguir:

- Clave primaria: Es la clave candidata que el usuario elegirá, por consideraciones ajenas al modelo relacional, para identificar las tuplas de la relación. Los atributos que forman parte de la clave primaria no pueden tomar valores nulos (reacuérdese las restricciones inherentes que es la *integridad de entidad*[19]).
- Claves alternativas: Son aquellas claves candidatas que no han sido elegidas como claves primarias de la relación.

2.3.5.3 Clave ajena

Una clave ajena[20] de una relación *R2* es un conjunto no vacío de atributos cuyos valores han de coincidir con los valores de la clave primaria de una relación *R1* (*R1* y *R2* pueden ser la misma relación). Se dice que *R2* es la relación que *referencia*, mientras que *R1* es la relación *referenciada*.

---

[19] *Establece que: "Ningún atributo que forme parte de la clave primaria puede tomar valores nulos".*

[20] *Son denominadas también "externas" o "foráneas".*

En la figura 2.5 se muestra un ejemplo de clave ajena, donde valores del atributo Cod_prog de la relación CURSO_DOCTORADO (relación que referencia) deben coincidir con los que es la clave primaria de la relación PROGRAMA (relación referenciada). *[ADOR00]*

PROGRAMA

Cod_prog	*Nombre*	*Departamento*
123FG	Ing. Informática	Leng. y Sistemas
123FH	Derechos fundam.	Derecho soc.
458TG	Documentación	Biblioteconomía

CURSO_DOCTORADO

Cod_curso	*Nombre*	*N_horas*	*Cod_prog*
DF000012	Sociología de los dchos. Fun.	25	123FH
DF000021	Teoría jurídica	30	123FH
D0000034	Evaluación de revistas cient.	20	458TG
II0000087	Almacenes de datos OLAP	100	123FG
II0000142	Evaluación procesos SW	25	123fg
D0000487	Automatización bibliotecas	35	458TG

FIGURA 2.5 Ejemplo de clave ajena

2.3.5.4 Integridad referencial

A menudo se desea asegurar que un valor que aparece en una relación para un conjunto de atributos determinado aparezca también en otra relación para un cierto conjunto de atributos. Esta condición se denomina *integridad*[21] *referencial*.

Dentro de las restricciones propias del modelo relacional se encuentra la de integridad referencial, que es de condición y acción específicas y que afirma:"Si una relación *R2*

[21] *El término* ***Integridad*** *se refiere a la exactitud o corrección de los datos en la base de datos.*

tiene un descriptor *CA* que es clave ajena que referencia a la clave primaria *CP* de la relación *R1*, todo valor de *CA* debe coincidir con un valor de *CP*, o ser nulo". Esta es la condición de la restricción, la cual puede expresarse como un predicado:

$$R2.CA = R1.CP$$

Los descriptores *CA* y *CP* han de ser definidos sobre el mismo dominio y se permite que sobre *CA* se defina, si es necesario, la restricción de obligatoriedad (si no se define, la clave ajena admitirá valores nulos[22]).

En cuanto a la acción, es de tipo específico (aunque no es de rechazo en todos los casos). Si se intenta insertar una tupla en la tabla que referencia R2, que no cumpla la condición, la acción es de rechazo. Si la condición falla debido a una operación de borrado de tuplas o de modificación de la clave primaria en la tabla referenciada R1, existe en SQL la posibilidad de elegir entre cuatro opciones, tanto para la operación de borrado como para la de modificación:

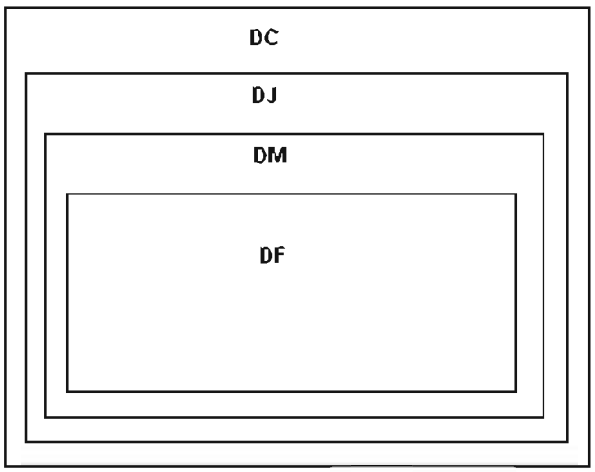
- NO ACTION (rechazar la operación)
- CASCADE (propagar la modificación o borrar las tuplas de la tabla que referencia)
- SET NULL (poner valor nulo en la CA de la tabla de referencia)
- SET DEFAULT (poner valor por defecto en la CA de la tabla que referencia)

La primera se toma por omisión y supone el rechazo de la correspondiente operación de borrado o de modificación, la cual no llega a llevarse a cabo. En la segunda se borran (o modifican) todas las tuplas de la tabla que referencia cuyo valor de la clave ajena coincide con el valor de la clave primaria que se borra (o se modifica). En la opción SET NULL se pone valor nulo en todos los atributos que forman parte del descriptor que constituye la clave ajena, cuyo valor ha desaparecido en la clave primaria de la tabla referenciada. En el último caso se pone el valor que se ha definido por defecto para la clave ajena en la tabla que referencia. ***[ADOR00]***

[22] *La clave ajena en la tabla que REFERENCIA es la que admitiría valores nulos, nunca la clave primaria de la tabla REFERENCIADA.*

2.4 TIPOS DE DEPENDENCIAS

Existen distintos tipos de dependencias: funcionales, multivaluadas, jerárquicas y de combinación (también llamadas producto). Cada tipo de dependencia se caracteriza por ser una asociación particular entre los datos. Además, cada tipo de dependencia constituye un caso particular del grupo que le sigue; así, las dependencias funcionales son un caso particular de las dependencias multivaluadas así sucesivamente.



DF (Dependencias Funcionales)
Como caso particular de:

DM (Dependencias Multivaluadas)
Como caso particular de:

DJ (Dependencias Jerárquicas)
Como caso particular de:

DC (Dependencias de Combinación)

FIGURA 2.6 Distintos tipos de dependencias

Aun cuando cada tipo de dependencia tenga una serie de características particulares, todas ellas presentan aspectos comunes:

- Son invariantes en el tiempo, siempre que no varíe el mundo real que se está modelando; por lo tanto, se han de cumplir para cualquier extensión del esquema de relación.
- Son propiedades inherentes al contenido semántico de los datos y forman parte de las restricciones de usuario del modelo relacional.
- A partir de una extensión válida de un esquema de relación si será posible comprobar que una dependencia no se cumple para ese esquema, ya que las dependencias se van a cumplir siempre, para cualquier extensión.

El grupo más restringido (y también más numeroso) de asociación entre los datos es el de las dependencias funcionales. Sobre este conjunto de dependencias se apoyan las tres primeras formas normales y la forma normal de Boyce-Codd. ***[ADOR00]***

2.4.1 Dependencia funcional

Las dependencias funcionales desempeñan un papel fundamental en la diferenciación entre los buenos diseños de bases de datos y los malos. Una dependencia funcional es un tipo de restricción que constituye una generalización del concepto de clave.

Definición: Sea el esquema de relación *R(A, DF)*, y sean *X, Y* dos subconjuntos de *A*, a los que llamamos descriptores. Se dice que *Y* depende funcionalmente de *X* o que *X* implica o determina[23] a *Y*, y se denota $X \rightarrow Y$ si, y solo si, a cada valor de **x** del atributo *X*, le corresponde un único valor **y** del atributo *Y*.

Un ejemplo de dependencia funcional es la que existe entre los atributos *DNI* de un profesor y *Nombre* del profesor:

$$DNI\_profesor \rightarrow Nombre\_profesor$$

Si además cumpliera que no pueden existir dos profesores con el mismo nombre, esto implicaría que el nombre también puede actuar como clave de la relación Profesor, y por lo tanto, también determina funcionalmente a su DNI, es decir:

$$Nombre\_profesor \longleftrightarrow DNI\_profesor$$

Cuando dos o más atributos se implican funcionalmente mutuamente se dice que son *equivalentes*. Una herramienta muy útil a la hora de explicar las dependencias funcionales es el grafo o diagrama de dependencias funcionales, mediante el cual se presenta un conjunto de atributos y las dependencias existentes entre ellos.

[23] *Un determinante o implicante es un conjunto de atributos del que depende funcionalmente otro conjunto de atributos al que se llama implicado.*

En el grafo aparecen los nombres de los atributos unidos por flechas, las cuales indican las dependencias funcionales y parten del implicante hacia el implicado. Cuando el implicante de una dependencia no es único atributo, es decir, se trata de un implicante compuesto, los atributos que lo componen se encierran en un recuadro y la flecha parte de éste, no de cada atributo.

En la figura 2.7 se presenta un ejemplo de cómo se visualizan las dependencias; podemos observar que *Cod_Estudiante* determina funcionalmente el *Nombre* y la *Dirección*, como indica la correspondiente flecha, de forma análoga, *Cod_Curso* determina el *Nombre* del curso, el *Num_Horas* y el *Programa*; mientras que en conjunto *Cod_Estudiante* y *Cod_Curso* (lo que se indica mediante el recuadro que lo incluye) determinan la *Fecha* y la *Nota*.

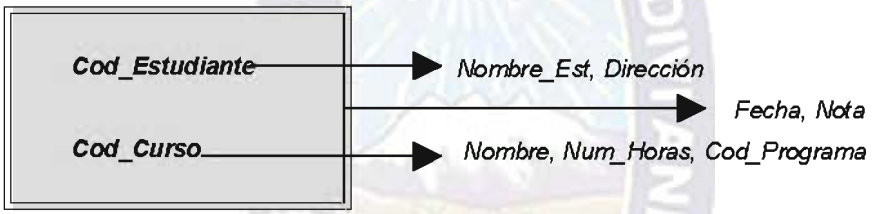


FIGURA 2.7 Ejemplo de diagrama de dependencias funcionales

Otra forma de representar un conjunto de dependencias funcionales aparece en la figura 2.8.

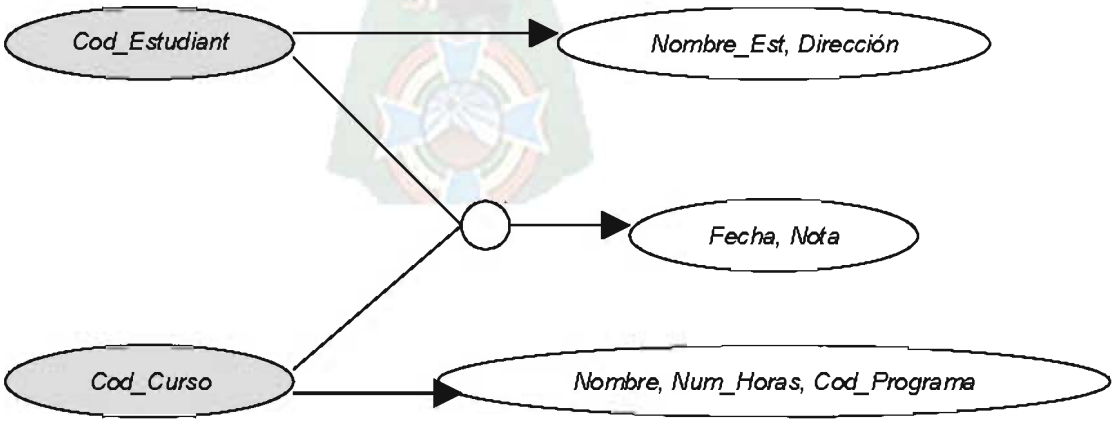


FIGURA 2.8 Ejemplo de otra forma de representar las dependencias funcionales

2.4.1.1 Dependencia funcional plena o completa

Definición: Se dice que *Y* tiene dependencia funcional completa o plena de *X*, si depende funcionalmente de *X*, pero no depende de ningún subconjunto de este.

Se representa por $X \Rightarrow Y$

Un ejemplo de dependencia funcional plena podría ser en la relación:

$$SE\_MATRICULA\ (Cod\_Curso, Cod\_Edición, Cod\_Estudiante, Nota)$$

Por tanto:

$$X \Rightarrow Y \text{ si } \nexists X' \subset X \setminus X' \rightarrow Y$$

Misma que refleja la nota que obtiene un estudiante en una edición de un curso (un curso puede tener varias ediciones); si suponemos que un estudiante puede matricularse en varias ediciones de un curso y en varios cursos distintos, y que, como es natural, en un curso se matriculan varios estudiantes se da la siguiente dependencia funcional:

$$Cod\_Curso, Cod\_Edición, Cod\_Estudiante \rightarrow Nota$$

Sin embargo:

$$\nexists X' (Cod\_Curso, Cod\_Edición, Cod\_Estudiante) \setminus X' \rightarrow Nota$$

por lo tanto, *Nota* depende funcionalmente de forma completa de *Cod_Curso*, *Cod_Edición*, *Cod_Estudiante*, esto es:

$$Cod\_Curso, Cod\_Edición, Cod\_Estudiante \Rightarrow Nota$$

lo que intuitivamente se puede interpretar como Nota constituye una información sobre el conjunto de curso, edición y estudiante; pero esta información no atañe a un estudiante o a una edición de un curso por separado.

En el ejemplo de la figura 2.7, la dependencia:

$$Cod\_Estudiante, Cod\_Curso \rightarrow Cod\_\text{Pr}ogama$$

No es plena, ya que:

$$Cod\_Curso \rightarrow Cod\_\text{Pr}ogama$$

Por tanto:

$$Cod\_Estudiante, Cod\_Curso \not\rightarrow Cod\_\text{Pr}ograma$$

Se dice que, en esta dependencia, *Cod_Estudiante* es un atributo redundante o ajeno a la dependencia, también llamado extraño.

2.4.1.2 Dependencia funcional trivial

Definición: Una dependencia funcional $X \rightarrow Y$ se dice que es trivial si Y es un subconjunto de X ($Y \subseteq X$). Por ejemplo, serán triviales las siguientes dependencias:

$$Cod\_Estudiante \rightarrow Cod\_Estudiante$$
$$Cod\_Curso, Cod\_Edición \rightarrow Cod\_Curso$$

2.4.1.3 Dependencia funcional elemental

Definición: Decimos que una dependencia funcional $X \rightarrow Y$ es elemental si Y es un atributo único no incluido en X, y no existe X' incluido en X tal que $X' \rightarrow Y$, es decir, la dependencia funcional elemental es una dependencia funcional completa, no trivial y en la que el implicado es un atributo único:

$X \rightarrow Y$ es elemental si $(\nexists Y' \subset Y) \wedge (Y \subseteq X) \wedge (\nexists X' \subset X \setminus X' \rightarrow Y)$

No todas las dependencias funcionales son útiles para la teoría de la normalización, sino solamente las elementales, que acabamos de definir, y que son un subconjunto de las dependencias funcionales.

2.4.1.4 Dependencia funcional transitiva

Definición: Dado el esquema de relación R(X, Y, Z) en las que existen las siguientes dependencias funcionales:

$$X \rightarrow Y \; ; \; Y \rightarrow Z \; ; \; Y \not\rightarrow X$$

Se dice que *Z* tiene una ***dependencia transitiva*** respecto a *X* a través de *Y*, lo que se representa por

$$X \dashrightarrow Z$$

Si, además, $Z \not\rightarrow Y$ se dice que la dependencia transitiva es ***estricta***.

Si consideramos la relación:

CURSO_PROGRAMA ($Cod\_Curso, Cod\_\mathrm{Pr}ograma, Cod\_Departamento$)

En donde tenemos para cada curso su código, el programa que lo incluye y el departamento del que depende el programa (suponiendo que un curso se imparte en un único programa y que un programa lo prepara un único departamento) se tendrán las siguientes dependencias funcionales:

$$Cod\_Curso \rightarrow Cod\_\mathrm{Pr}ograma$$
$$Cod\_\mathrm{Pr}ograma \rightarrow Cod\_Departamento$$

Además:

$Cod\_\mathrm{Pr}ograma \not\rightarrow Cod\_Curso$ (ya que en un programa se imparten varios cursos)

Entonces:

$$Cod\_Curso \rightarrow Cod\_Departamento$$

por tanto, la dependencia funcional entre *Cod_Curso* y *Cod_Departamento* es una dependencia transitiva a través de *Cod_Programa*, que se representa por:

$$Cod\_Curso \dashrightarrow Cod\_Departamento$$

(lo que se puede interpretar intuitivamente como que *Cod_Curso* es una información sobre el curso, pero indirectamente, ya que constituye una información sobre el programa y éste, a su vez, sobre el curso).
Además, como $Cod\_Departamento \not\rightarrow Cod\_Programa$, se da una dependencia transitiva estricta.

En cambio, si tuviéramos la relación:

CURSO ($Cod\_Curso, Nombre, Cod\_Programa$)

Con las siguientes dependencias:

$$Cod\_Curso \rightarrow Nombre$$
$$Nombre \rightarrow Cod\_Curso$$
$$Cod\_Curso \rightarrow Cod\_Programa$$
$$Nombre \rightarrow Cod\_Oprama$$

Ninguna de las dependencias:

$$Cod\_Curso \rightarrow Cod\_Programa \text{ ni } Nombre \rightarrow Cod\_Programa$$

Son transitivas, al ser los atributos *Cod_Curso* y *Nombre* equivalentes.

2.4.2 Dependencias multivaluadas

Definición: Sean X e Y dos descriptores. ***X multidetermina a Y*** si para cada valor de X existe un conjunto bien definido de valores posibles en Y, con independencia del resto de los atributos de la relación (Fagin, 1977).

Notación: $X \rightarrow\rightarrow Y$

Por ejemplo, en la figura 2.9 el atributo *Nombre* multidetermina a titulación:

Nombre	*Titulación*
Felipe	Magisterio Música
Felipe	Ing. Superior en Informática
Antonia	Ing. Caminos

FIGURA 2.9 Ejemplo de una tabla con dependencia multivaluada

Al igual que ocurría con las dependencias funcionales, las dependencias multivaluadas también pueden ser de varios tipos, pero no se entrara en detalle, por el momento.

Existe un tipo especial de dependencias multivaluadas, que so las dependencias multivaluadas jerárquicas.

2.4.3 Dependencias jerárquicas

Definición: Sean X, Y y Z descriptores. X multidetermina jerárquicamente a Y y Z si en la proyección de los atributos X multidetermina a Y y Z multidetermina a Z.

Notación: $X \rightarrow\rightarrow Y \setminus Z$

Un ejemplo de este tipo de dependencias se muestra en la figura 2.10, donde una persona posee varios títulos y trabaja en varios proyectos, donde no se exige ninguna titulación para trabajar en ningún proyecto (no hay relaciones entre estos atributos):

Nombre	*Titulación*	*Proyecto*
Felipe	Magisterio en Música	A
Felipe	Ing. Superior en Informática	B
Antonia	Ing. Caminos	A

FIGURA 2.10 Ejemplo de una tabla con dependencia jerárquica

2.4.4 Dependencias en combinación

Definición: Una relación R tiene dependencias en combinación respecto de sus proyecciones (R1, R2, ... Rn) si R = R1 * R2 * ... * Rn.

Notación: $DJ*(R1,R2,...,Rn)$

Un ejemplo de esta propiedad es el que aparece en la figura 2.11, donde existe una relación de "envidia" entre dos personas. Si, por ejemplo Felipe trabaja en varios proyectos (A y B) y además realiza todos los cursos posibles sobre Bases de Datos e Ingeniería de software (las dos primeras tuplas). Si en esos momentos se inserta una nueva tupla (la tercera) donde Antonia trabaja en el proyecto B y realiza un curso de BD, entonces forzosamente Felipe, que trabaja en el mismo proyecto, ha de recibir el mismo curso:

Nombre	*Tema_Curso*	*Proyecto*
Felipe	BD	A
Felipe	IS	B
Antonia	BD	B
Felipe	BD	B

FIGURA 2.10 Ejemplo de una tabla con dependencias en combinación ***[ADOR00]***

2.5 NORMALIZACIÓN DE BASES DE DATOS

En primera instancia se entenderá por normalización como el proceso mediante el cual se transforman datos complejos a un conjunto de estructuras de datos más pequeñas, 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.

La normalización también hace las cosas fáciles de entender. Los seres humanos tenemos la tendencia de simplificar las cosas al máximo. Lo hacemos con casi todo, desde los animales hasta con los automóviles. Vemos una imagen de gran tamaño y la hacemos más simple agrupando cosas similares juntas. Las guías que la normalización provee crean el marco de referencia para simplificar una estructura de datos compleja.

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 mucho menor uso de espacio en disco.

El proceso de normalización tiene un nombre y una serie de reglas para cada fase. Esto puede parecer un poco confuso al principio, pero poco a poco se va entendiendo el proceso, así como las razones para hacerlo de esta manera. ***[MYSQ03]***

Observe la figura 2.11, para entender de mejor manera el criterio de repetición de datos, mas comúnmente denominado ***redundancia*** de datos.

VCP

V#	*CIUDAD*	*P#*	*CANT.*
V1	Londres	P1	300
V1	Londres	P2	200
V1	Londres	P3	400
V1	Londres	P4	200
V1	Londres	P5	100
V1	Londres	P6	100
V2	París	P1	300
V2	París	P2	400
V3	París	P2	200
V4	Londres	P2	200
V4	Londres	P4	300
V4	Londres	P5	400

FIGURA 2.11 Valor del ejemplo de la varrel VCP

Para ser específicos, cada tupla de *VCP* para el proveedor *V1* nos dice que *V1* está ubicado en *Londres*, cada tupla del proveedor *V2* nos dice que *V2* está ubicado en *París*, etcétera. De manera más general, el hecho de que un proveedor determinado esté ubicado en una ciudad dada es enunciado tantas veces como los envíos que hay de ese proveedor. A su vez, esta redundancia conduce a varios problemas adicionales. Por ejemplo, después de una actualización, el proveedor *V1* podría aparecer como si estuviera ubicado en Londres, de acuerdo con una tupla, y ubicado en *Ámsterdam*, de acuerdo con otra. De modo que tal vez un buen principio de diseño sea "cada hecho en un lugar" (es decir, evitar la redundancia). *El tema de la normalización adicional es en esencia sólo una formalización de ideas sencillas como ésta*, pero una formalización que sí tiene una aplicación muy práctica para el problema del diseño de bases de datos.

Por lo que respecta al modelo relacional, las relaciones siempre están normalizadas[24]. En cuanto a las *varrels*, podemos decir que también están normalizadas (siempre y cuando sus valores válidos sean relaciones normalizadas); por lo tanto, en lo que respecta al modelo relacional, también las *varrels* están siempre normalizadas. ***[DATE01]***

2.5.1 Primera forma normal (1FN)

La primera de las formas normales que se van a estudiar, la primera forma normal, impone un requisito muy elemental a las relaciones; a diferencia de las demás formas normales, no exige información adicional como las dependencias funcionales.

Un dominio es atómico si se considera que los elementos del dominio son unidades individuales. Se dice que el esquema de una relación R esta en la primera forma normal (1FN) si los dominios de todos los atributos de R son atómicos. ***[KORT02]***

Poner la base de datos en la Primera Forma Normal resuelve el problema de los encabezados de columna múltiples. Muy a menudo, los diseñadores de bases de datos inexpertos harán algo similar a la tabla no normalizada. Una y otra vez, crearán columnas que representen los mismos datos. La normalización ayuda a clarificar la base de datos y a organizarla en partes más pequeñas y más fáciles de entender. En lugar de tener que

[24] *De hecho, una forma de pensar en la disciplina de la normalización es la siguiente: la normalización nos ayuda a estructurar la base de datos de tal forma que las actualizaciones de tuplas individuales sean más aceptables lógicamente de lo que serían de otro modo (es decir, si el diseño no estuviese totalmente normalizado).*

entender una tabla gigantesca y monolítica que tiene muchos diferentes aspectos, sólo tenemos que entender los objetos pequeños y más tangibles, así como las relaciones que guardan con otros objetos también pequeños. ***[MYSQ03]***

En la figura 2.12 se puede observar que si un estudiante solicita más de una beca se tienen grupos repetitivos, y para pasar a 1FN habrá que repetir el resto de atributos de la tupla para cada uno de los valores del grupo repetitivo.

Esquema: ***ESTUDIANTE*** (*Código*, *Nombre*, *Curso*)

Código	***Nombre***	***Curso***
178263782	Pedro Perales	ERASMUS HIMMPA
031928733	Alberto González	MÉTRICA ERASMUS
763459374	Francisco Vidal	HIMMPA MÉTRICA

Es una tabla pero no una relación.

NO ESTA EN 1FN

Hay grupos repetidos

Código	***Nombre***	***Curso***
178263782	Pedro Perales	ERASMUS
178263782	Pedro Perales	HIMMPA
031928733	Alberto González	MÉTRICA
031928733	Alberto González	ERASMUS
763459374	Francisco Vidal	HIMMPA
763459374	Francisco Vidal	MÉTRICA

ESTA EN 1FN.

FIGURA 2.12 1FN y grupos repetitivos

Definición: Se dice que una relación está en 1FN cuando cada atributo sólo toma un valor del dominio simple subyacente. ***[ADOR00]***

2.5.2 Dificultades en el diseño de bases de datos relacionales

Antes de continuar con el estudio de las formas normales hay que examinar lo que puede salir mal en un mal diseño de bases de datos. Entre las propiedades indeseables que puede tener un mal diseño están:

- Repetición de la información
- Imposibilidad de la representación de determinada información

Estos problemas se estudiarán con ayuda de un diseño de bases de datos modificado para el ejemplo bancario: Supóngase que la información relativa a los préstamos se guarda en una sola relación, *empréstito*, que se define mediante el esquema de relación:

$$\textit{Esquema-empréstito} = (\begin{matrix} nombre-sucursal, ciudad-sucursal, activo, \\ nombre-cliente, número-préstamo, importe \end{matrix})$$

La figura 2.13 muestra un ejemplo de la relación *empréstito* (*esquema-empréstito*). Cada tupla *t* de la relación empréstito tiene el siguiente significado intuitivo.

- t[$activo$] es el volumen de activo de la sucursal denominada t[$nombre-sucursal$]
- t[$ciudad-sucursal$] es la ciudad en la que se ubica la sucursal denominada t[$nombre-sucursal$]
- t[$número-préstamo$] es el número asignado al préstamo concedido por la sucursal denominada t[$nombre-sucursal$] al cliente llamado t[$nombre-cliente$]
- t[$importe$] es el importe del préstamo cuyo número es t[$número-préstamo$]

nombre-sucursal	*ciudad-sucursal*	*activo*	*nombre-cliente*	*número-préstamo*	*importe*
Centro	Arganzuela	9.000.000	Santos	P-17	1.000
Moralzarzal	La Granja	2.100.000	Gómez	P-23	2.000
Navacerrada	Aluche	1.700.000	López	P-15	1.500
Centro	Arganzuela	9.000.000	Sotoca	P-14	1.500
Becerril	Aluche	400.000	Santos	P-93	500
Collado Mediano	Aluche	8.000.000	Abril	P-11	900
Navas de la Asunción	Alcalá de Henares	300.000	Valdivieso	P-29	1.200
Segovia	Cerceda	3.700.000	López	P-16	1.300
Centro	Arganzuela	9.000.000	Gonzáles	P-18	2.000
Navacerrada	Aluche	1.700.000	Rodríguez	P-25	2.500
Galapagar	Arganzuela	7.100.000	Amo	P-10	2.200

FIGURA 2.13 Relación empréstito de ejemplo

Supóngase que se desea añadir un nuevo préstamo a la base de datos. Digamos que el préstamo se lo concede la sucursal de Navacerrada a la señora Fernández por un importe de 1500. Sea el $número - préstamo$ P-31.

En el diseño hace falta una tupla con los valores de todos los atributos de *Esquema-empréstito*. Por tanto, hay que repetir los datos del activo y de la ciudad de la sucursal de Navacerrada y añadir la tupla a la relación *empréstito*. En general, los datos del activo y de la ciudad deben aparecer una vez para cada préstamo concedido por esa sucursal.

La repetición de la información en el diseño alternativo no es deseable. La repetición de la información desaprovecha el espacio. Además, complica la actualización de la base de datos. Supóngase, por ejemplo, que el activo de la sucursal de Navacerrada cambia de 1.700.000 a 1.900.000. Con el diseño original hay que modificar una tupla de la relación *sucursal*. Con el diseño alternativo hay que modificar muchas tuplas de la relación empréstito. Por tanto, las actualizaciones resultan más costosas con el diseño alternativo que con el original. Cuando se lleva a cabo la actualización en la base de datos alternativa hay que asegurarse de que se actualicen todas las tuplas correspondientes a la sucursal de Navacerrada, o la base de datos mostrará dos valores diferentes del activo para esa sucursal.

Otro problema del diseño *Esquema-empréstito* es que no se puede representar de manera directa la información relativa a cada sucursal

$(nombre-sucursal, ciudad-sucursal, activo)$ a menos que haya como mínimo un préstamo en esa sucursal. Esto se debe a que las tuplas de la relación empréstito exigen los valores de $número-préstamo, importe$ y $nombre-cliente$. ***[KORT02]***

2.5.3 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 (también denominada clave 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. Podemos insertar un registro sin un exceso de datos en la mayoría de las tablas. ***[MYSQ03]***

También está basada en el concepto de dependencia plena y en las interrelaciones existentes entre los atributos principales (que se encuentran en alguna de las claves) y no principales (que no se encuentran en ninguna clave) de una relación.

Definición: Se dice que una relación está en 2FN si:

- Está en 1FN.
- Cada atributo no principal tiene dependencia funcional completa respecto de cada una de las claves.

La segunda forma normal no se cumple cuando algún atributo no principal depende funcionalmente de algún subconjunto de la clave.

Se puede afirmar que cualquier relación binaria se encuentra siempre en 2FN; así como también está en 2FN cualquier relación en la que todas las claves son simples, es decir, contienen un solo atributo. Asimismo, está en 2FN cualquier relación en la que todos sus atributos son ***principales***, es decir, forman parte de alguna clave.

Siempre es posible transformar un esquema de relación que no esté en 2FN, en esquemas de relación en 2FN, sin que se produzca pérdida de información ni de dependencias.

Sea el esquema relacional *ESTUDIANTE _ BECA* (AT, DEP)[25] donde:

AT= $\{Cod\_Estudiante, Cod\_Beca, Fecha\_Sol, Titulo\}$

DEP = $\{^{Cod\_Estudiante, Cod\_Beca \rightarrow Fecha\_Sol}_{Cod\_Estudiante \rightarrow Titulo}\}$

Que refleja las becas que solicitan los estudiantes, la fecha en que lo han hecho y la titulación del estudiante.

La clave de la relación *ESTUDIANTE _ BECA* es $(Cod\_Estudiante, Cod\_Beca)$, se puede observar que el atributo $Titulo$ no es un hecho (una información) acerca de la totalidad de la clave, sino acerca de parte de ella (en este caso del atributo $Cod\_Estudiante$). Esta relación ***no está en 2FN***.

Transformamos la relación *ESTUDIANTE_BECA* en las relaciones *ESTUDIANTE_BECA1* y *ESTUDIANTE* que ya ***sí se encuentran en 2FN. [ADOR00]***

ESTUDIANTE_BECA1 (AT1, DEP1), donde:

AT1= $\{Cod\_Estudiante, Cod\_Beca, Fecha\_Sol\}$

DEP1 = $\{Cod\_Estudiante, Cod\_Beca \rightarrow Fecha\_Sol\}$

Ahora *ESTUDIANTE* (AT2, DEP2), donde:

AT2 = $\{Cod\_Estudiante, Titulo\}$

DEP2 = $\{Cod\_Estudiante \rightarrow Titulo\}$

[25] *Recordemos que AT son los descriptores o un conjunto de atributos que existirán en la relación y DEP son las dependencias que existen entre los descriptores.*

2.5.4 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. Comentamos anteriormente 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 deben haber datos repetidos. Esto provee un esquema limpio y elegante, que es fácil de trabajar y expandir. ***[MYSQ03]***

La tercera forma normal (3FN) está basada en el concepto de dependencia transitiva.

Definición: Un esquema de relación R está en tercera forma normal si, y sólo sí:

- Está en 2FN.
- No existe ningún atributo no principal que dependa transitivamente de alguna clave de R.

La tercera forma normal no se cumple cuando existen atributos no principales que dependen funcionalmente de otros atributos no principales.

Se puede afirmar que toda relación binaria se encuentra automáticamente en 3FN, así como toda relación cuyos atributos son todos principales, o bien cuando hay un único atributo no principal.

Siempre es posible transformar un esquema de relación que no está en 3FN, en esquemas de relación en 3FN, sin que se produzca pérdida de información ni de dependencias funcionales.

Sea el esquema de relación *ESTUDIANTE* (AT, DEP) donde:

AT = $\{Cod\_Estudiante, Cod\_Proyecto, Nombre\_Proyecto\}$

DEP = $\{\begin{matrix} Cod\_Proyecto \rightarrow Nombre\_Proyecto \\ Cod\_Estudiante \rightarrow Cod\_Proyecto \end{matrix}\}$

La única calve del esquema de relación es el atributo $Cod\_Estudiante$. El atributo $Nombre\_Proyecto$ es un hecho acerca del atributo $Cod\_Proyecto$, atributo que no forma parte de la clave. Por lo tanto, este esquema de relación no está en 3FN (está en 2FN).

Se puede transformar la relación *ESTUDIANTE* en las relaciones *ESTUDIANTE1* y *PROYECTO* que sí se encuentran ya en 3FN.

ESTUDIANTE1 (AT1, DEP1) donde:

AT1 = $\{Cod\_Estudiante, Cod\_Proyecto\}$

DEP1 = $\{Cod\_Estudiante \rightarrow Cod\_Proyecto\}$

PROYECTO (AT2, DEP2) donde:

AT2 = $\{Cod\_Proyecto, Nombre\_Proyecto\}$

DEP2 = $\{Cod\_Proyecto \rightarrow Nombre\_Proyecto\}$ ***[ADOR00]***

2.5.5 Forma normal de Boyce-Codd (FNBC)

Las tres formas normales que acabamos de exponer fueron las propuestas originalmente por CODD (1970), pero, como ya hemos señalado, con el paso del tiempo se mostraron insuficientes para afrontar ciertos problemas en relación que presentaban varias claves candidatas compuestas que se solapaban. Por ello, en 1974, Boyce y Codd definieron la llamada formal normal que lleva su nombre (FNBC), ya descrita por Heath, aunque con ligeras diferencias, en 1971. Se trata de una redefinición más estricta de la 3FN.

Se dice que una relación se encuentra en FNBC si, y sólo si, todo determinante es una clave candidata.

Como ejemplo, se puede hacer referencia a lo que ocurre en una universidad, si consideráramos que los nombres de los cursos no pueden repetirse, y que un estudiante obtiene una calificación en cada curso al que asiste, la relación:

$$ASISTE\,(Cod\_Curso, Nom\_Curso, Cod\_Estudiante, Calificación)$$

Tendría las siguientes dependencias funcionales:

$$Cod\_Curso \leftrightarrow Nom\_Curso$$
$$Cod\_Curso, Cod\_Estudiante \rightarrow Calificación$$

Y por tanto, tendría dos claves candidatas: $\{Cod\_Curso, Cod\_Estudiante\}$ y $\{Nom\_Curso, Cod\_Estudiante\}$. Esta relación está en 3FN (todos sus atributos, menos uno, son principales), sin embargo tiene anomalías de actualización, ya que se repetiría el nombre y el código de los cursos por cada estudiante que asistiese a ellos; el problema es debido a que la relación ASISTE no se encuentra en FNBC, ya que tanto Cod_Curso como Nom_Curso son determinantes, pero no son claves candidatas de la relación.

Si una relación cuyas claves no están solapadas se encuentra en 3FN está también en FNBC. Ahora bien, la existencia de claves candidatas solapadas no lleva siempre consigo que la relación no esté en FNBC. Consideremos el esquema:

$$SE\_MATRICULA1\,(Cod\_Curso, Cod\_Edición, Cod\_Estudiante, Fecha)$$

Donde:

$$Cod\_Curso, Cod\_Edición, Cod\_Estudiante \rightarrow Fecha$$
$$Cod\_Edición, Cod\_Estudiante, Fecha \rightarrow Cod\_Curso$$

Las claves candidatas[26] de esta relación son $Cod\_Curso, Cod\_Edición, Cod\_Estudiante$ y $(Cod\_Edición, Cod\_Estudiante, Fecha)$, que se solapan ya que comparten los atributos

[26] *Recordemos que una clave candidata de una relación es un conjunto no vacío de atributos (descriptor) que identifican unívocamente y minimamente cada tupla de una relación.*

$Cod\_Edición$ y $Cod\_Estudiante$; sin embargo, debido a que los únicos determinantes son los dos descriptores anteriores, que son claves candidatas, la relación si se encuentra en FNBC. ***[ADOR00]***

Se puede afirmar que toda relación binaria está en FNBC.

Definición: Se dice que una relación se encuentra en FNBC si y sólo si todo determinante es clave candidata.

Por ejemplo, la relación $R(\{A,B,C,D\},\{A \longleftrightarrow B; A,C \rightarrow D\})$ no se encuentra en FNBC, ya que los determinantes A y B no son claves candidatas, sino que lo son los conjuntos *{A,C}* y *{B,C}*.

Sin embargo, las relaciones $R1(\{A,B\},\{A \longleftrightarrow B\})$ y $R2(\{A,C,D\},\{A,C \rightarrow D\})$ sí se encuentran en FNBC.

2.5.6 Formas normales superiores

Hasta aquí se ha estudiado las ideas de normalización e incluso la forma normal de Boyce-Codd (que es hasta donde puede llevarnos el concepto de dependencia funcional). Ahora se completara la explicación examinando la cuarta y quinta formas normales (4FN y 5FN). Es en la cuarta forma normal en la que ya se hace uso de una nueva clase de dependencia, denominada dependencia multivaluada, misma que se ha estudiado a detalle antes. En la quinta forma normal, se hace uso de la definición de dependencia junta[27], misma que es una generalización de las dependencias multivaluadas.

Estas formas normales tratan acerca de hechos con valores múltiples. Un hecho de valores múltiples puede corresponder a una relación muchos-a-muchos, como con empleados y destrezas, o a una relación muchos-a-una, como con los hijos de un empleado (asumiendo sólo un padre como empleado). Por "muchos-a-muchos" nosotros entendemos que un empleado puede tener varias destrezas y/o una destreza puede pertenecer a varios empleados. Note que vemos a la relación muchos-a-uno entre hijos y

[27] *También denominada dependencia jerárquica.*

padres como un hecho de un valor único para el hijo pero como un hecho de valores múltiples para el padre.

En un sentido, las formas normales cuarta y quinta corresponden también a claves compuestas. Estas formas normales intentan minimizar el número de campos envueltos en una clave compuesta, como es sugerido por el ejemplo siguiente.

Bajo la cuarta forma normal, un tipo de registro no debería contener dos o más hechos multivalor independientes acerca de una entidad. En adición, el registro debe satisfacer la tercera forma normal. ***[UNAL]***

2.5.6.1 Cuarta forma normal (4FN)

Teorema (Fagin). Sea *R {A, B, C}* una varrel, donde *A*, *B* y *C* son conjuntos de atributos. Entonces R es igual a la junta de sus proyecciones sobre *{A, B}* y *{A, C}* si y solamente si R satisface las dependencias multivaluadas $A \twoheadrightarrow B \setminus C$.
Entonces, se dice que una relación se encuentra en 4FN si y sólo si las únicas dependencias multivaluadas no triviales son aquellas en las que una clave multidetermina un atributo.

Por ejemplo, la relación $R(\{A,B,C\},\{A \twoheadrightarrow B; A, A \twoheadrightarrow C\})$ no se encuentra en 4FN, ya que su clave candidata es el conjunto *{A, B, C}* y en la primera dependencia el atributo C no participa y en la segunda es el atributo B el que no participa.

Sin embargo, las relaciones $R1(\{A,B\},\{A \twoheadrightarrow B\})$ y $R2(\{A,C\},\{A \twoheadrightarrow C\})$ sí se encuentran en 4FN.

Definición: La varrel *R* está en 4FN si y solamente si siempre que existan subconjuntos *A* y *B* de los atributos de *R*, tales que las dependencias multivaluadas no trivial[28] $A \twoheadrightarrow B$

[28] *Considerando que una dependencia multivaluada $A \twoheadrightarrow B$ es **trivial** si A es un superconjunto de B o la unión de A y B es el encabezado completo.*

se satisfaga, entonces todos los atributos de R son también dependientes funcionalmente de *A*.

2.5.6.2 Quinta forma normal (5FN)

Se dice que una relación se encuentra en 5FN si y sólo sí:

- Se encuentra en 4FN.
- Toda dependencia de combinación está implicada por una clave candidata.

También se puede dar otra definición para la 5FN: "Una relación se encuentra en 5FN si y sólo si toda dependencia funcional, multivaluada o de combinación no trivial es consecuencia de las claves candidatas".

Definición: Una varrel R está en 5FN – también llamada forma normal de proyección-junta (FN/PJ) – si y solamente sí cada dependencia de junta non trivial[29] valida para R está implicada por las claves candidatas de R.

2.5.6.3 Otras formas normales

La cuarta forma normal no es, de ningún modo, la forma normal "definitiva". Hay tipos de restricciones denominadas dependencias de reunión que generalizan las dependencias multivaloradas y llevan a otra forma normal denominada **forma normal de reunión por proyección (FNRP)**, denominada también quinta forma normal, esto en algunos libros. Hay una clase de restricciones todavía más generales, que lleva a una forma normal denominada **forma normal de dominios y claves (FNDC)**.

Un problema práctico del empleo de estas restricciones generalizadas es que no sólo es difícil razonar con ellas, sino que tampoco hay un conjunto de reglas de inferencia segura y completa para razonar sobre las restricciones. Por tanto, la FNRP y la forma normal de dominios y claves se utilizan muy raramente.

---

[29] *La dependencia junta {A, B,..., Z} es trivial si y solamente sí una de las proyecciones A, B,..., Z es la proyección identidad de R (es decir, la proyección sobre todos los atributos de R).*

2.6 PRINCIPIO DE DISEÑO ORTOGONAL

En esta sección examinamos brevemente otro principio del diseño de bases de datos, uno que no parte de la normalización en *sí* pero que se asemeja a ella debido a que es *científico*. Se llama el principio de diseño ortogonal. Considere la figura 2.14, la cual muestra un diseño para proveedores obviamente malo pero posible; en ese diseño, la varrel *VA* corresponde a los proveedores que están ubicados en París, la varrel *VB* corresponde a los proveedores que no están ubicados en París o que tienen un status mayor que 30 (es decir, de manera general, los predicados de varrel). Como la figura indica, el diseño conduce a ciertas *redundancias*; para ser específicos, la tupla del proveedor *V3* aparece dos veces, una en cada varrel.

/* proveedores en París */

VA

V#	***PROVEEDOR***	***STATUS***	***CIUDAD***
V2	Jones	10	París
V3	Blake	30	París

/* proveedores que no están en París */

VB

V#	PROVEEDOR	STATUS	CIUDAD
V1	Smith	20	Londres
V3	Blake	30	París
V4	Clark	20	Londres
V5	Adams	30	Atenas

FIGURA 2.14 Un mal diseño, aunque posible, para proveedores

Por cierto, observe que la tupla *debe* aparecer en ambos lugares. Si por el contrario, suponemos que aparece (digamos) en *VB* y no en *VA*. Entonces, aplicar la Suposición del Mundo Cerrado a *VA* nos diría que no se da el caso que el proveedor *V3* esté ubicado en París. Sin embargo, *VB* nos dice que *sí* se da el caso que el proveedor *V3* esté ubicado en París. En otras palabras, tendríamos entre las manos una contradicción y la base de datos sería inconsistente.

Por supuesto, el problema con la figura 2.14 es obvio: es precisamente el hecho de que es posible que una misma tupla aparezca en dos varrels distintas. En otras palabras las dos varrels tienen significados que se traslapan, en el sentido de que es posible que la misma tupla satisfaga los predicados de varrel de ambas.

2.6.1 El principio de diseño ortogonal (versión inicial)

Dentro de una base de datos dada, dos varrels no pueden tener significados que se traslapen[30].

Las ideas a destacar son:

1. Recuerde que desde el punto de vista del usuario, *todas* las varrels son varrels base (además de las vistas que están definidas como meras formas abreviadas). En otras palabras, el principio se aplica al diseño de todas las bases de datos "expresables", no sólo a la base de datos "real"; entra en acción una vez más el *principio de relatividad*[31] *de la base de datos*. (Desde luego, se aplican también observaciones similares a los principios de normalización).
2. Observe que no es posible que dos varrels tengan significados que se traslapen a menos que sean del mismo tipo (es decir, a menos que tengan el mismo encabezado).

Aún no terminamos con el principio del diseño ortogonal; existen refinamientos importantes que es necesario abordar. Observe la figura 2.15 que muestra otro diseño obviamente malo, pero posible, para proveedores. Aquí, las varrels por sí mismas no tienen significados que se traslapen, pero sus proyecciones sobre {*V#,PROVEEDOR*} en realidad si los tienen (de hecho, los significados de las dos proyecciones son idénticos). Como consecuencia, un intento por insertar la tupla – digamos – (V6,López) es una vista definida como la unión de esas dos proyecciones, hará que la tupla (V6,López,*t*) sea insertada en VX y la tupla (V6,López,*c*) en VY (donde *t* y *c* son valores predeterminados aplicables). Resulta claro que necesitamos ampliar el principio de diseño ortogonal para que se haga cargo de problemas como el de la figura 2.

[30] *Recordemos lo que se menciono en el Capitulo 1, "son significados traslapados cuando una misma tupla satisface los predicados de dos o más variables relacionales".*

[31] *El principio de relatividad se refiere a la diferencia que existe entre la base de datos real y la base de datos "expresable", la cual consiste en alguna combinación de varrels base y vistas. La base de datos expresable es manipulada por los usuarios de un determinado sistema.*

VX

V#	*PROVEEDOR*	*STATUS*
V1	Smith	20
V2	Jones	10
V3	Blake	30
V4	Clark	20
V5	Adams	30

VY

V#	*PROVEEDOR*	*CIUDAD*
V1	Smith	Londres
V2	Jones	París
V3	Blake	París
V4	Clark	Londres
V5	Adams	Atenas

FIGURA 2.15 Otro mal diseño, aunque posible, para proveedores

2.6.2 Observaciones

1. Suponga que comenzamos con nuestra varrel usual de proveedores V, pero por motivos de diseño decidimos dividirla en un conjunto de restricciones. Entonces, el principio de diseño ortogonal nos dice que las restricciones en esa división deben estar todas disjuntas, en el sentido de que jamás ninguna tupla de proveedores puede aparecer en más de una de ellas. Nos referimos a dicha división como una *descomposición ortogonal*. *Nota*: el término *ortogonalidad* se deriva del hecho de que lo que en realidad significa el principio de diseño, es que las varrels base deben ser mutuamente independientes (sin traslapar significados). Por supuesto, el principio proviene del sentido común, pero del sentido común *formalizado* (como los principios de la normalización).

2. El objetivo general del diseño ortogonal es reducir la redundancia y evitar las anomalías de actualización (de nuevo como la normalización). De hecho, complementa la normalización en el sentido de que – en general – la normalización reduce la redundancia *dentro* de las varrels, mientras que la ortogonalidad la reduce a *través* de las varrels.

3. La ortogonalidad podría lograrse mediante el sentido común, pero en la práctica a menudo se ignora (de hecho, a veces se recomienda hacerlo). Los diseños como el siguiente – de una base de datos financiera – son demasiado comunes:

   ACTIVIDADES_1997 { PARTIDA#, DESCRIPCION, IMPORTE, SALDO_NVO }

   ACTIVIDADES_1998 { PARTIDA#, DESCRIPCION, IMPORTE, SALDO_NVO }

   ACTIVIDADES_1999 { PARTIDA#, DESCRIPCION, IMPORTE, SALDO_NVO }

   ACTIVIDADES_2000 { PARTIDA#, DESCRIPCION, IMPORTE, SALDO_NVO }

   ACTIVIDADES_2001 { PARTIDA#, DESCRIPCION, IMPORTE, SALDO_NVO }

   De hecho, codificar significados dentro de nombres – de varrels o de cualquier otra cosa – viola el principio de información, el cual establece (para recordarlo) que toda la información en la base de datos debe ser expresada explícitamente en términos de valores y de ninguna otra forma.

2.7 ASPECTOS DE DISEÑO A CONSIDERAR

2.7.1 Vistas

Se ha visto que a partir de un conjunto de varrels[32] como DEPTO y EMP, y un conjunto de valores de relación de éstas, las expresiones relacionales nos permiten obtener otros valores de relación a partir de los ya dados (por ejemplo, juntando dos de las varrels dadas). Ahora, es momento de representar un poco más de terminología. A las varrels originales (dadas) se les denomina ***varrels base***[33] y a sus valores de relación se les llama y a sus valores de relación se les llama ***relaciones base***; a una relación que es o que puede ser obtenida a partir de dichas relaciones base por medio de alguna expresión relacional, se le denomina relación derivada o derivable.

---

[32] *Recordemos que varrel es una terminología que se maneja para representar a una variable relacional.*

[33] *También denominada varrels auténticas.*

Ahora bien, en primer lugar, los sistemas relacionales tienen obviamente que proporcionar un medio para crear las varrels base. Por ejemplo, en SQL esta tarea es realizada por medio de la instrucción *CREATE TABLE* (aquí, "*TABLE*" tiene un significado muy específico: el de varrel base). Y a estas varrels base obviamente se les tiene que dar un nombre; por ejemplo:

CREATE TABLE EMP ...;

Sin embargo, por lo regular los sistemas relacionales también manejan otro tipo de varrel con nombre, denominada **vista**, cuyo valor en cualquier momento dado es una relación *derivada* (de donde se puede imaginar una vista, a grandes rasgos, como una **varrel derivada**). El valor de una vista determinada en un momento dado, es cualquiera que sea el resultado de evaluar cierta expresión relacional en ese momento; dicha expresión relacional es especificada en el momento de que se crea la vista en cuestión. Por ejemplo, la instrucción:

CREATE VIEW EMPSUP AS
(EMP WHERE SALARIO > 33K) { EMP#, NOMEMP, SALARIO } ;

Esta instrucción podría ser usada para definir una vista llamada EMPSUP.

Cuando esta instrucción es ejecutada, la expresión relacional que sigue a AS – la expresión que define la vista – no es evaluada, sino que el sistema simplemente la "recuerda" de alguna manera (de hecho, guardándola en el catalogo bajo el nombre específico EMPSUP). Sin embargo para el usuario, ahora es como si en realidad fuese una varrel de la base de datos, denominada EMPSUP, con un valor actual como se indica (sólo) en las partes donde apuntan las flechas de la figura 2.16. Y el usuario debe poder operar sobre esa vista tal como si fuera una varrel base[34]. ***[DATE01]***

[34] *Si se piensa en DPTO y en EMP como varrels auténticas, entonces se podría pensar en EMPSUP como un varrel virtual; es decir, una varrel que aparentemente existe por derecho propio, pero que de hecho no existe (su valor, en cualquier momento dado, depende de los valores de otras varrels determinados).*

EMPSUP

EMP#	NOMEMP	DEPTO#	SALARIO
E1	López	D1	40K
E2	Cheng	D1	42K
E3	Pérez	D2	30K
E4	Hernández	D2	35K

FIGURA 2.16 EMPSUP como una vista de EMP
(columna y fila señalada por flechas)

Las vistas también sirven para la seguridad de la base de datos, en términos de que no es deseable que todos los usuarios puedan ver la totalidad del modelo lógico. Las consideraciones sobre la seguridad pueden exigir que algunos datos queden ocultos para los usuarios. Considérese una persona que necesita saber el número de préstamos de un cliente pero que no necesita ver el importe del préstamo. Esta persona debería ver una relación descrita en el algebra relacional mediante:

$$\Pi_{nombre-cliente,número-préstamo,nombre-sucursal}(prestatario \triangleright\triangleleft préstamo)$$

A parte de las consideraciones sobre la seguridad puede que se desee crear un conjunto personalizado de relaciones que se adapte mejor que el modelo lógico a la intuición de un usuario concreto. Por ejemplo, puede que un empleado del departamento de publicidad quiera ver una relación que conste de los clientes que tengan abierta una cuenta o concedido un préstamo en el banco y de las sucursales con las que trabajan. La relación que se crearía para ese empleado es:

$$\Pi_{nombre-sucursal,nombre-cliente}(impositor \triangleright\triangleleft cuenta) \cup$$
$$\Pi_{nombre-sucursal,nombre-cliente}(prestatario \triangleright\triangleleft préstamo)$$

Se puede trabajar con gran número de vistas sobre cualquier conjunto dado de relaciones reales.

2.7.1.1 Definición de vistas

Las vistas se definen utilizando la instrucción create view. Para definir una vista hay que darle un nombre e indicar la consulta que la va a calcular. La forma de instrucción create view es:

create view v as <expresión de consulta>

Donde <expresión de consulta> es cualquier expresión legal de consulta del álgebra relacional. El nombre de la vista se representa mediante v.

Como ejemplo considérese la vista consistente en las sucursales y sus clientes. Supóngase que se desea que esta vista se denomine *todos-los-cliente*. Esta vista se define de la manera siguiente:

create view *todos-los-clientes* as

$$\Pi_{nombre-sucursal,nombre-cliente}(impositor \rhd\lhd cuenta) \cup$$
$$\Pi_{nombre-sucursal,nombre-cliente}(prestatario \rhd\lhd préstamo)$$

Una vez se ha definido una vista se puede utilizar el nombre de la vista para referenciar a la relación virtual que genera la vista. Utilizando la vista *todos-los-clientes* se puede averiguar el nombre de todos los clientes de la sucursal de Navacerrada escribiendo, esto como ejemplo.

$$\Pi_{nombre-cliente}(\sigma_{nombre-sucursal="Navacerrada"}(todos-los-clientes))$$

Los nombres de las vistas pueden aparecer en cualquier lugar en el que pueda encontrarse el nombre de una relación, siempre y cuando no se ejecuten sobre las vistas operaciones de actualización.

La definición de las vistas se diferencia de la operación asignación del álgebra relacional. Supóngase que se define la relación *r1* de la manera siguiente:

$$r1 \leftarrow \Pi_{nombre-sucursal, nombre-cliente}(impositor \rhd\lhd cuenta)$$
$$\cup \Pi_{nombre-sucursal, nombre-cliente}(prestatario \rhd\lhd préstamo)$$

La operación de asignación se evalúa una vez, y *r1* no cambiará cuando se actualicen las relaciones *impositor*, *cuenta*, *préstamo* o *prestatario*. En cambio, si hay alguna modificación en estas relaciones, el conjunto de tuplas de la vista todos-los-clientes también cambia. De manera intuitiva, en cualquier momento dado, el conjunto de tuplas de la evaluación de la expresión de consulta que define en ese momento la vista.

Por tanto, si una relación de vistas se calcula y se guarda, puede quedar desfasada si las relaciones utilizadas para definirla se modifican. En vez de eso, las vistas suelen implementarse de la manera siguiente. Cuando se define una vista, el sistema de base de datos guarda la definición de la próxima vista, en vez del resultado de la evaluación de la expresión de consulta guardada. Por tanto, la relación de vistas vuelve a calcularse siempre que se evalúa la consulta.

Algunos sistemas de bases de datos permiten que se guarden las relaciones de vistas, pero se aseguran de que, si las relaciones reales utilizadas en la definición de la vista cambian, la vista se mantenga actualizada. Estas vistas de denominan ***vistas materializadas***. El proceso de mantener actualizada la vista se denomina ***mantenimiento de vistas***. Las aplicaciones en las que se utilizan frecuentemente una vista se benefician del uso de vistas materializadas, igual que las aplicaciones que demandan una rápida respuesta a ciertas consultas basadas en las vistas. Las ventajas de la materialización de una vista para consultas deben sopesarse frente a los costes de almacenamiento y la sobrecarga añadida por las actualizaciones.

2.7.1.2 Actualizaciones mediante vistas y valores nulos

Aunque las vistas son una herramienta útil para las consultas, plantean problemas significativos si con ellas se expresan las actualizaciones, las inserciones o los borrados. La dificultad radica en que las modificaciones de la base de datos expresadas en términos de vistas deben traducirse en modificaciones de las relaciones reales en el modelo lógico de la base de datos.

Para ilustrar el problema considérese un empleado que necesita ver todos los datos de préstamos de relación *préstamo* salvo *importe*. Sea *préstamo-sucursal* la vista dada al empleado. Se define esta vista como:

Create view préstamo-sucursal as

$$\Pi_{nombre-sucursal,número-préstamo}(préstamo)$$

Dado que se permite que los nombres de las vistas aparezcan en cualquier parte en la que estén permitidos los nombres de relaciones, el empleado puede escribir:

$$préstamo-sucursal \leftarrow préstamo-sucursal \cup \{(P-37,"Navacerrada")\}$$

Esta inserción debe representarse mediante una inserción en la relación *préstamo*, dado que *préstamo* es la relación real a partir de la cual se genera la vista *préstamo-sucursal*. Sin embargo, para insertar una tupla en *préstamo* hay que tener algún valor para *importe*. Hay dos enfoques razonables para trabajar con esta inserción:

- Rechazar la inserción y devolver al usuario un mensaje de error.
- Insertar una tupla (P-37, "Navacerrada", nulo) en la relación *préstamo*.

2.7.2 Tabulación cruzada

El análisis estadístico suele necesitar el agrupamiento de varios atributos. Considérese una aplicación en que una tienda desea averiguar las prendas que son más populares. Supóngase que las prendas están caracterizadas por su nombre de artículo, su color y su talla y que se tiene la relación *ventas* con el *esquema ventas (nombre-artículo, color, talla, número)*. Supóngase que *nombre-artículo* puede adoptar los valores (falda, vestido, camisa, pantalón), *color* puede adoptar los valores (oscuro, pastel, blanco) y *talla* puede adoptar los valores (pequeña, mediana, grande).

Dada una relación utilizando para el análisis de datos se puede identificar algunos de sus atributos como ***atributos de medida***, ya que miden algún valor y pueden agregarse. Por ejemplo, el atributo *número* de la relación *ventas* es un atributo de medida, ya que mide la

cantidad de unidades vendidas. Algunos de los demás atributos (o todos ellos) de la relación se identifican como ***atributos de dimensión***, ya que definen las dimensiones en las que se ven los atributos de medida. En la relación *ventas*, *nombre-artículo*, *color* y *talla*, son atributo de dimensión[35].

Para analizar los datos multidimensionales puede que el administrador desee ver los datos dispuestos como se ven en la tabla de la figura 2.17. La tabla muestra las cifras totales de diferentes combinaciones de *nombre-artículo* y *color*. El valor de *talla* se especifica como todas, lo que indica que los valores mostrados son un resumen para todos los valores de *talla*.

La tabla de la figura 2.17, es un ejemplo de tabulación cruzada, también denominada tabla dinámica. En general las tabulaciones cruzadas son tablas en las que los valores de uno de los atributos (por ejemplo, A) toman las cabeceras de las filas, los valores del otro atributo (por ejemplo, B) forman las cabeceras de las columnas y los valores de cada celda se obtienen como sigue: cada celda puede identificarse como (a_i, b_j), donde a_i es un valor de A y b_j un valor de B. Si hay como máximo una tupla con cualquier valor de (a_i, b_j), el valor de la celda se obtiene de esa única tupla (si es que hay alguna); por ejemplo, puede ser el valor de uno o varios atributos de la tupla. Si puede haber varias tuplas con el valor (a_i, b_j), el valor de la celda debe obtenerse por agregación de las tuplas con ese valor. En este ejemplo la agregación utilizada es la suma de los valores del atributo *número*. En este ejemplo la tabulación cruzada también tiene una columna y una fila adicionales que guardan los totales de las celdas de cada fila o columna. La mayor parte de las tabulaciones cruzadas tiene esas filas y columnas de resumen.

Las tabulaciones cruzadas son diferentes de las tablas relacionales que suelen guardarse en las bases de datos, ya que el número de columnas de la tabulación cruzada depende de los datos. Una modificación en los valores de los datos puede dar lugar a que se añadan más columnas, lo que no resulta deseable para el almacenamiento de los datos. No obstante, la vista de tabulación cruzada es deseable para mostrársela a los usuarios.

---

[35] *Considerando que una versión mas realista de la relación ventas tendría más dimensiones, como tiempo o lugar de ventas, y más medidas como el valor monetario de la venta.*

talla: **all**

nombre - artículo	*color*			
	oscuro	***pastel***	***blanco***	***Total***
falda	8	35	10	53
vestido	20	10	5	35
camisa	14	7	28	49
pantalón	20	2	5	27
Total	62	54	48	164

FIGURA 2.17 Tabulación cruzada de ventas con nombre-artículo y color

La representación de las tabulaciones cruzadas sin valores resumen en un formulario relacional con un número fijo de columnas es directa. La tabulación cruzada son columnas o filas resumen puede representarse introduciendo el valor especial **all** (todos) para representar los subtotales, como en le figura 2.18. La norma SQL: 1999 utiliza realmente el valor **null** (nulo) en lugar de **all**; pero, para evitar confusiones con los valores nulos habituales, en el libro seguirá utilizando **all**.

nombre-articulo	*color*	*número*
falda	oscuro	8
falda	pastel	35
falda	blanco	10
falda	**all**	53
vestido	oscuro	20
vestido	pastel	10
vestido	blanco	5
vestido	**all**	35
camisa	oscuro	14
camisa	pastel	7
camisa	blanco	28
camisa	**all**	49
pantalones	oscuro	20
pantalones	pastel	2
pantalones	blanco	5
pantalones	**all**	27
all	oscuro	62
all	pastel	54
all	blanco	48
all	**all**	164

FIGURA 2.18 Representación relacional de los datos de la figura 2.17

Considérense las tuplas (falda, **all**, 53) y (vestido, **all**, 53). Se han obtenido estas tuplas eliminando las tuplas individuales con diferentes valores de *color* y sustituyendo el valor de *número* por un agregado, es decir, una suma. El valor **all** puede considerarse representante del conjunto de los valores de un atributo. Las tuplas con el valor all sólo para la dimensión *color* pueden obtenerse mediante una consulta de SQL que lleve a cabo una agrupación en la columna *nombre-artículo*. De manera parecida, se puede utilizar una agrupación de *color* para conseguir las tuplas con el valor **all** para *nombre-artículo*, y una agrupación sin atributo alguno (que en SQL puede omitirse simplemente) puede utilizarse para obtener la tupla con el valor all para *nombre-artículo* y *color*.

La generalización de las tabulaciones cruzadas, que son bidimensionales, a n dimensiones puede visualizarse como n dimensionales, denominados cubos de datos.

El sistema de procesamiento analítico en línea o sistema OLAP es un sistema interactivo que permite a los analistas ver diferentes resúmenes de los datos multidimensionales. Las palabras en línea indican que los analistas deben poder solicitar nuevos resúmenes y obtener respuestas en línea, en pocos segundos, y no deberían verse obligados a esperar mucho tiempo para ver el resultado de las consultas. ***[KORT02]***

2.8 MÉTRICAS PARA LA EVALUACIÓN DE LA COMPLEJIDAD DE BASES DE DATOS RELACIONALES

Es bien sabido que las métricas son importantes, ya que el objetivo de estas son para que se puedan evaluar los productos de software, en términos de calidad; las cuales son ampliamente difundidas por la Organización Internacional de Estandarización (ISO[36], 1994). Existen factores que se deben controlar, uno de estos es la mantenibilidad, uno de los más conflictivos ya que supone entre el 60 y el 80 por ciento de los costes que se tienen que considerar en el ciclo de vida del sistema. Está reconocido que las métricas del software son un buen medio para entender, monitorizar, controlar, predecir y probar el desarrollo software y los proyectos de mantenimiento (Briand et al., 1996). En resumen las métricas despachan resultados numéricos, mismos que sirven para una buena toma de desiciones.

[36] *La ISO (Internacional Standarization Organization) es la entidad internacional encargada de favorecer la normalización en el mundo. Con sede en Ginebra, es una federación de organismos nacionales, estos, a su vez, son oficinas de normalización que actúan de delegadas en cada país. Se creo para dar más eficiencia a las normas nacionales.* ***[BIBL98]***

Tres son los factores que influyen en la mantenibilidad: entendibilidad, modificabilidad y testeabilidad, los cuales a su vez están influenciados por la complejidad (Li y Chen, 1987). Sin embargo, una métrica general para medir la complejidad es como el "santo grial" (Fenton, 1994). Henderson-Sellers (1996) divide la complejidad en tres: computacional, psicológica y representacional, y para la psicológica define tres componentes: complejidad del problema, factores cognitivos humanos y complejidad del producto, en estas últimas se concentra este estudio.

2.8.1 Métricas propuestas

Desde que a finales de los sesenta el Dr. Codd propusiera su modelo relacional (Codd, 1970), se ha intensificado la investigación en el campo de las bases de datos y los productos de bases de datos relacionales han generado una importante industria.

El único indicador utilizado para medir la calidad de una base de datos relacional fue la teoría de la normalización, a partir de la cual Gray et al. (1991) propone un ratio de normalidad.

Para probar los beneficios que se tienen con el modelo que se presenta en este trabajo, se considerara las métricas propuestas en un artículo, que precisamente lleva como título *"Métricas para la Evaluación de la complejidad de Bases de Datos relacionales"*, la cual es planteada por el Grupo ALARCOS, los cuales tienen origen Español. Considera este grupo de estudio presentar cinco métricas que se clasifican en dos categorías: métricas orientadas a tablas y métricas orientadas a esquemas.

Dentro de las orientadas a tablas se propone:

- NUMERO DE ATRIBUTOS (NA): NA es el número de atributos en todas las tablas del esquema y NA(A) es el número de atributos de la tabla A.

- GRADO DE REFERENCIABILIDAD (REFERENTIAL DEGREE RD): RD se define como el número de claves ajenas del esquema relacional y RD(A) es el número de claves ajenas de la tabla A.

Dentro de las orientadas a esquemas se propone:

- PROFUNDIDAD DEL ARBOL REFERENCIAL (DEPTH REFERENTIAL TREE DRT): DRT se define como la longitud del máximo camino referencial del esquema relacional. Los ciclos sólo se consideran una vez.

- RATIO DE NORMALIDAD (NORMALITY RATIO NR): NR se define como el número de tablas en tercera forma normal o superior dividido entre el número de tablas en el esquema, lo que significa;

$$NR = \frac{NT3FN}{NTS}$$

Donde NT3FN número de tablas en 3FN (o superior) y NTS es el número de tablas en el esquema.

En la figura 2.19 se dará un ejemplo, práctico para poder aplicar las métricas señaladas.

```
CREATE TABLE EMPLEADO
        (
                NOMBREP VARCHAR(15) NOT NULL,
                INIC CHAR,
                APELLIDOVARCHAR(15) NOT NULL,
                NSS CHAR(9) NOT NULL,
                FECHAEN         DATE,
                DIRECCION VARCHAR(30)
                SEXO CHAR,
                SALARIO DECIMAL(10,2),
                NSSSUPER CHAR(9),
                ND INT  NOT NULL,


CONSTRAINT CLPEMP
                PRIMARY KEY (NSS),
CONSTRAINT CLESUPEREMP
FOREIGN KEY (NSSSUPER) REFERENCES EMPEADO(NSS)
        ON DELETE SET NULL ON UPDATE CASCADE,
CONSTRAINT CLEDEPTOEMP
FOREIGN KEY (ND) REFERENCES DEPARTAMENTO  (NUMEROD)
        ON DELETE SET DEFAULT ON UPDATE CASCADE
        ),
```

```
CREATE TABLE PROYECTO
        (
                NOMBREPR VARCHAR(15)    NOT NULL,
                NUMEROPR INT    NOT NULL,
                LUGARPR VARCHAR(15),
                NUMD    INT     NOT NULL,
        PRIMARY KEY (NUMEROP),
        UNIQUE (NOMBREP),
FOREIGN KEY (NUMD) REFERENCES DEPARTAMENTO (NUMEROD)
        );



CREATE TABLE TRABAJA_EN
        (
                NSSE    CHAR(9) NOT NULL,
                NUMP    INT     NOT NULL,
                HORAS   DEDIMAL(3,1)    NOT NULL,
        PRIMARY KEY (NSSE,NUMP),
FOREIGN KEY (NSSE) REFERENCES EMPLEADO (NSS),
FOREIGN KEY (NUMP) REFERENCES PROYECTO (NUMEROP)
        ),
```

```
CREATE TABLE DEPARTAMENTO
        (
                NOMBRED  VARCHAR(15) NOT NULL,
                NUMEROD  INT  NOT NULL,
                NSSGTE   CHAR(9)   NOT NULL,
                FECHAININGTE DATE,
COSNTRAINT CLPDEPTO  PRIMARY KEY (NUMEROD),
CONSTRAINT DLSDEPTO  UNIQUE(NOMBRED),
CONSTRAINT CLEGTEDEPTO
FOREIGN KEY (NSSGTE) REFERENCES EMPLEADO(NSS)
        ON DELETE SET DEFAULT ON UPDATE CASCADE
        ),
```

```
CREATE TABLE DEPENDIENTE
        (
                NSSE      CHAR(9)   NOT NULL,
                NOMBRE_DEPEND   VARCHAR(15)      NOT
NULL,
                SEXO      CHAR,
                FECHAAN DATE,
                RELACION          VARCHAR(8),
        PRIMARY KEY (NSSE, NOMBRE_DEPEND
        ),

CREATE TABLE LUGAR_DEPTS
        (
                NUMEROD  INT  NOT NULL,
                LUGARD   VARCHAR(15)  NOT NULL,
        PRIMARY KEY(NUMEROD,LUGARD),
FOREIGN KEY (NUMEROD) REFERENCES DEPARTAMENTO
(NUMEROD)
        ON DELETE CASCADE ON UPDATE CASCADE  ),
```

FIGURA 2.19 Ejemplo de una base de datos relacional

En la siguiente figura 2.19 se presenta el grafo relacional para el ejemplo anterior, donde las flechas indican las relaciones de integridad referencial:

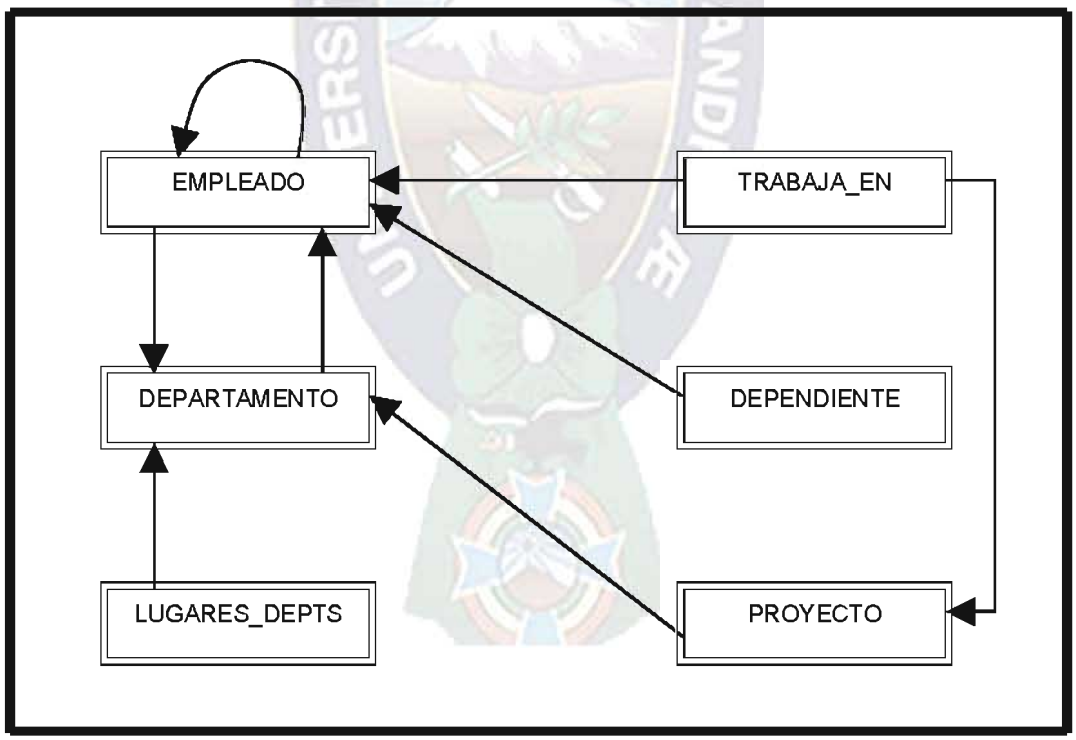


FIGURA 2.20 Esquema del ejemplo anterior

Los valores de las métricas correspondientes del ejemplo se muestran en la siguiente tabla:

	NA	RD	DRT	NR
EMPLEADO	10	2		
DEPARTAMENTO	4	1		
LUGARES_DEPTS	2	1		
PROYECTO	4	1		
TRABAJA_EN	3	2		
DEPENDIENTE	5	1		
ESQUEMA	28	8	5	1

En la tabla, podemos observar todos los valores obtenidos para las métricas a partir de la información que nos da tanto el ejemplo como su esquema. Por ejemplo, el valor para la métrica DRT es cinco, que se obtiene del camino referencial siguiente:

$$TRABAJA\_EN \rightarrow PROYECTO \rightarrow DEPARTAMENTO \rightarrow EMPLEADO \rightarrow EMPLEADO \rightarrow DEPARTAMENTO$$

En este punto conviene recordar que, por definición, la métrica DRT sólo considera los ciclos una vez.

2.8.2 Medición del modelo lógico

Se dará a continuación un resumen de lo que hasta ahora se ha dicho, y algunos otros parámetros de medición que también son importantes.

METRICA	NOTACIÓN	DEFINICIÓN
Número de atributos de una tabla	NA(T) (Number of Attributes)	Definida como el número de atributos de una tabla T
Número de Claves Ajenas	NFK(T) (Number of foreign keys)	Definida como el numero de claves ajenas de una tabla T
Profundidad del Árbol referencial de una Tabla	DRT (T) (Depth of the referential tree)	Definida como la profundidad máxima de todos los caminos referenciales del grafo que se forma, tomando la tabla T como el nodo raíz del grafo y todas las tablas relacionadas con T mediante integridad referencial como el resto de nodos y siendo las relaciones de integridad referencial los arcos del mismo

Ratio de Claves Ajenas de una Tabla	RFK(T) (Ratio of foreign Key)	Definida como el porcentaje de atributos de la tabla T que son claves ajenas $$RFK(T) = \frac{NFK(T)}{NA(T)}$$
Número de Tablas	NT (Number of Tables)	Definida como el número de tablas que hay en el esquema
Cohesión del esquema	COS (Cohesion of the Schema)	Definida como la suma del número de tablas al cuadrado que hay en cada componente no conexa del grafo del esquema, siendo los nodos de este grafo las tablas del esquema y los arcos las relaciones de integridad referencial $$\cos = \sum_{i=1}^{US} NT_{US}$$
Ratio de Normalidad	NR (Normalita Ratio)	Definida como la relación entre el número de tablas en tercera forma normal (o superior) entre el número total de tablas $$NR = \frac{NT3FN}{NTS}$$ Siendo NT3FN es el número de tablas en 3FN
Número de Atributos	NA (Numbers of Attributes)	Definida como el número total de atributos que hay en el esquema $$NA = \sum_{i=1}^{NT} NA(T_i)$$
Número de Claves Ajenas	NFK (Number of Foreign Keys)	Definida como el número total de claves ajenas que hay definidas en el esquema $$NFK = \sum_{i=1}^{NT} NFK(T_i)$$
Profundidad del árbol referencial	DRT (Depth of the Referential Tree)	Definida como la profundidad máxima de todos los caminos referenciales del grafo que se forma tomando las tablas del esquema como los nodos y las relaciones de integridad como los arcos del mismo $$DRT = \max_{i=1}^{NT}(DRT(T_i))$$
Ratio de Claves Ajenas	RFK (Ratio of Foreign Key)	Definida como el porcentaje de atributos del esquema que son claves ajenas $RFK = \frac{NFK}{NA}$

FIGURA 2.21 Parámetros de medición para bases de datos relacionales

Capitulo 3

CONSTRUCCIÓN DEL MODELO PROPUESTO PARA LA OPTIMIZACIÓN DEL DISEÑO DE BASES DE DATOS RELACIONALES

3.1 INTRODUCCIÓN

En este capitulo se manejara terminología y definiciones que servirán específicamente para la definición del modelo. En primera instancia se hará una interpretación de lo que es el principio de diseño ortogonal, ya que esta da una pauta general para evitar que existan variables relacionales expresables con significados que llegan a traslaparse.

El estudio de vectores en el plano servirá para poder definir el modelo y poder complementar la definición del principio de diseño ortogonal.

Para la definición del modelo se ha visto por conveniente realizar los casos que se consideraran más propensos a darse. Por lo tanto en el presente capitulo se procederá a desarrollar el modelo propuesto de optimización para el diseño de bases de datos relacionales.

3.2 PRINCIPIO DE DISEÑO ORTOGONAL

Es importante tener en cuenta este tipo de diseño, en vista de que el objetivo del mismo es la reducción de redundancia de información y básicamente en su teoría menciona un conjunto de divisiones que se necesitan realizar a una varrel determinada. Además, para el planteamiento del modelo, se considera importante el término ortogonal, en términos de que se tendría la restricción de la existencia de independencia de varrels.

Definición.- Sean *A* y *B* dos varrels base cualesquiera de la base de datos. Entonces, no deben existir descomposiciones sin perdida de *A* y *B* en *A1, … , Am* en *B1, … , Bn* (respectivamente), tales que alguna proyección *Ai* en el conjunto *A1, … , Am* y alguna proyección *Bj* en el conjunto *B1, … , Bn* tengan significados que se traslapen.

Las ideas a destacar son:

1. Aquí, el término "descomposición sin pérdida" significa exactamente lo mismo de siempre; es decir, la descomposición en un conjunto de proyecciones tales que:

   - La varrel dada pueda ser reconstruida juntando de nuevo las proyecciones;
   - Ninguna de esas proyecciones sea redundante en el proceso de reconstrucción.

2. Esta versión del principio incluye a la versión original, ya que una descomposición sin pérdida que existe siempre para la varrel *R* es la proyección identidad de *R* (es decir, la proyección sobre todos sus atributos).

3.3 VECTORES EN EL PLANO

Para la definición de vectores en el plano, es importante entender primero, como preámbulo dos conceptos, como lo son las magnitudes escalares y las magnitudes vectoriales. Ahora bien, estas definiciones y las posteriores servirán para entender de mejor manera el modelo de optimización para el diseño de bases de datos relacionales.

3.3.1 Magnitudes escalares

Las magnitudes cuyas cantidades quedan perfectamente determinadas al indicarse la medida y la unidad como las longitudes, las superficies, los volúmenes, las capacidades, se llaman magnitudes escalares. Por lo que, conceptos físicos que para su determinación requieren de una sola cantidad se denominan magnitudes escalares.

3.3.2 Magnitudes vectoriales

Los conceptos que para su determinación, requieren más de una cantidad, se llaman magnitudes vectoriales (Ej. La velocidad, la fuerza, etc.)

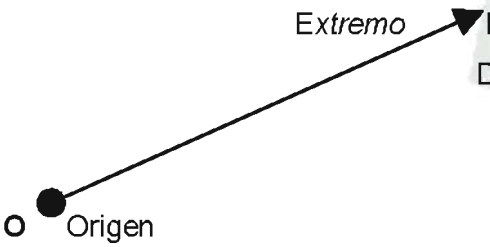
Todas las propiedades vectoriales, pueden obtenerse analíticamente, a partir de las siguientes definiciones de vectores: V^2 (en dos dimensiones en el plano). Un vector en el plano Coordenado Real, es un par ordenado de números reales.

La notación usual de un vector es: $\vec{A} = (a_1, a_2)$

Donde: a_1, a_2 son el primer y el segundo componente respectivamente del vector $\vec{A}$

3.3.3 Representación gráfica de los vectores

Geométricamente un vector se representa como un segmento orientado:



Donde:

O es el origen, **P** es el Extremo del vector.
La longitud **OP** representa el MODULO.
La inclinación, representa la DIRECCIÓN.
La posición de la flecha, indica el SENTIDO.
Por tanto un vector posee, Módulo, Dirección y Sentido.

Para graficar un vector, sobre un sistema de coordenadas, se hace coincidir su origen con el origen de coordenadas.

Su extremo coincidirá con un punto de coordenadas equivalentes a las componentes de vector. Así se establece una relación entre vector y punto.

Ejemplo 1: Representar geométricamente en el plano al vector: $\vec{A}$ = (3,1)

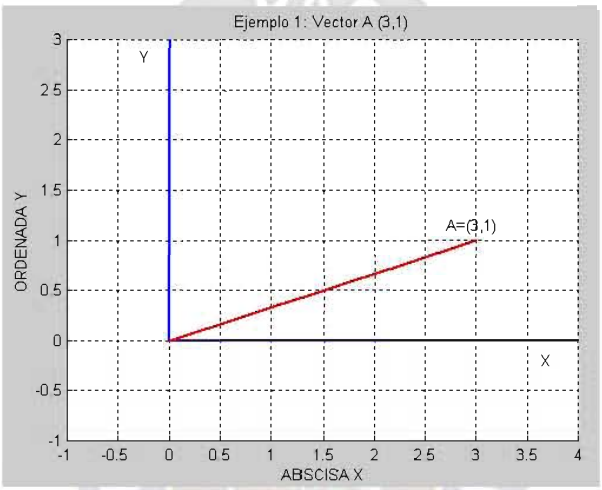


FIGURA 3.1 Ejemplo gráfico de un vector

Los componentes son: a_1 =3; a_2 =1 (Representado por la línea de color ROJO)

Como se puede observar en la figura 3.1, se coloca el origen del vector en el Punto (0,0), y el extremo en el punto que indica los datos del vector (3,1).

Por lo que, es así como se expresa geométricamente el vector.

Ejemplo 2: Representar geométricamente en el plano, a los vectores:

$\vec{A}$ = (5,1); $\vec{B}$ = (-2,4)

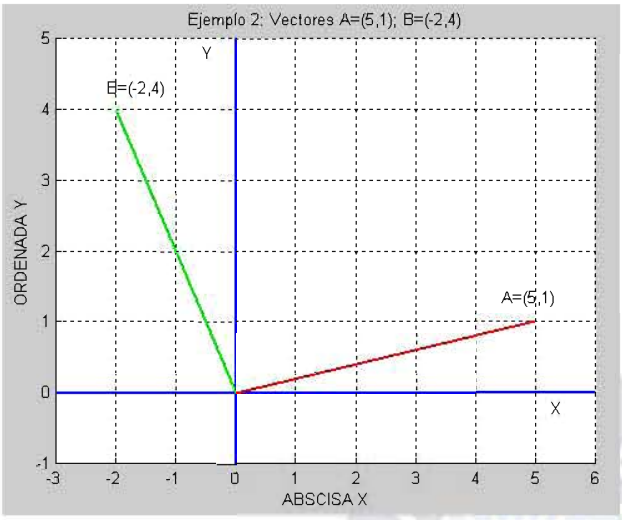


FIGURA 3.2 Ejemplo gráfico de la representación de dos vectores

Analizando, se tiene que el vector $\vec{A}$ = (5,1), donde a_1 = 5; a_2 = 1 (representado por la línea de color ROJO) y respectivamente el vector $\vec{B}$ = (-2,4) (representado por la línea de color verde), con los parámetros b_1 = -2; b_2 = 4.

3.3.4 Módulo de un vector

El modulo de un vector, (o longitud de segmento orientado), se determina por:

$$\text{Si } \vec{A} = (a_1, a_2) \Rightarrow |\vec{A}| = \sqrt{a_1^2 + a_2^2}$$

3.3.5 Producto escalar

El producto escalar (o producto punto, o producto interior); es otra operación entre vectores, cuyo resultado es un escalar, se define y se denota de la siguiente manera:

$$\text{Si: } \vec{A} = (a_1, a_2); \vec{B} = (b_1, b_2) \Rightarrow \vec{A} \circ \vec{B} = (a_1, a_2) \circ (b_1, b_2) = a_1 b_1 + a_2 b_2$$

Las principales propiedades del producto escalar son:

T1. $\vec{A} \circ \vec{B} = \vec{B} \circ \vec{A}$

T2. $(k\vec{A}) \circ \vec{B} = k(\vec{B} \circ \vec{A})$

T3. $\vec{A} \circ (\vec{B} + \vec{C}) = \vec{A} \circ \vec{B} + \vec{A} \circ \vec{C}$

T4. $\vec{A} \circ \vec{A} = |\vec{A}|^2; \quad Si: \vec{A} \circ \vec{A} = \vec{0} \Rightarrow \vec{A} = \vec{0}$

T5. $\vec{A} \circ \vec{B} = |\vec{A}||\vec{B}|Cos\theta; \quad \theta = Angulo(\vec{A}, \vec{B})$

T6. $\vec{A} \circ \vec{B} = 0 \Rightarrow \vec{A} \perp \vec{B}; Si: \vec{A} \neq \vec{0}; \vec{B} \neq \vec{0}$

3.3.6 Proyección ortogonal

Todo vector $\vec{A}$ puede expresarse en términos de otro vector $\vec{B}$ (Si: $\vec{A}, \vec{B} \in V^3$)

$\vec{A} = s\vec{B} + t\vec{B}^{\perp}$; Donde: $s = \dfrac{\vec{A} \circ \vec{B}}{|\vec{B}|^2}; \quad t = \dfrac{\vec{A} \circ \vec{B}^{\perp}}{|\vec{B}^{\perp}|^2}; \quad \vec{B}, \ \vec{B}^{\perp} \neq \vec{0}; \ s, t \in R$

3.4 ANÁLISIS DE LOS CASOS POSIBLES

Los casos que describo a continuación son los que considero más propensos a darse, en relación al proceso que se da cuando se plantea el modelo conceptual, para luego pasarlo a un modelo lógico, considerando en ese proceso la cardinalidad y la respectiva normalización de los esquemas. Cabe señalar que el análisis que se plantea en cada uno de los casos, se da para los dos esquemas relacionales planteados para su respectiva demostración.

3.4.1 Casos a ser considerados por el modelo

Caso 1: Es posible tener este diseño de los esquemas, teniendo en cuenta que la varrel R1, significa todos los proveedores que se encuentran en una determinada ciudad, mientras que R2, significa todos los proveedores que no están en dicha ciudad o cuentan con un Status mayor a 30.

R1 (V#, Proveedor, Status, Ciudad)

R2 (V#, Proveedor, Status, Ciudad)

Como se puede observar, ambas son verdaderas y necesarias, pero cabe la posibilidad que existan contradicciones, por lo que la base de datos seria también inconsistente. Lo que estaría ocurriendo es que se da el traslapado de la información.

Caso 2: En este caso las dos varrels no tienen significados que se traslapen, pero sus proyecciones sobre (V#, Proveedor), en realidad si los tienen (de hecho los significados de las dos proyecciones son idénticos).

R1 (V#, Proveedor, Status)
R2 (V#, Proveedor, Ciudad)

3.4.2 Casos a no ser considerados por el modelo

No serán considerados por el modelo, en vista de que los esquemas son considerados mutuamente independientes. Lo que quiere decir que no tienen elementos en común entre ellas.

Caso 3: Es la representación de dos esquemas relacionales que no tienen ningún tipo de proyección.

R1 (Cod_Docente, Nombre_Docente, Direccion_Docente)
R2 (Cod_Estudiante, Nombre_Estudiante, Materia_Cursar)

3.5 MODELO DE OPTIMIZACIÓN PARA EL DISEÑO DE BASES DE DATOS RELACIONALES

Para el desarrollo del modelo que evita el traslapado de información, se vio por conveniente representar a los esquemas relacionales en forma de vectores, para poder así demostrar el tipo de traslapado que se llegaría a dar. Entonces, se iniciara el análisis del modelo definiendo los componentes y las restricciones que deberán cumplir, por lo que se tiene:

$\vec{EA}$ = (a, b, c) $\vec{EB}$ = (d, e, f)

Donde:

- **a:** Número total de atributos que componen al esquema A.
- **b, e:** Número de atributos que coinciden comparando dos esquemas relacionales (en este caso el esquema A y el esquema B)
- **d:** Número total de atributos que componen el esquema B.
- **c, f:** Número de elementos que no han sido considerados por b, e.

Ahora bien, para que el modelo sea considerado criterioso, existen algunas restricciones que son importantes, las mismas que versan lo siguiente:

$$\mathbf{0 \leq c + f \leq 3}$$

$$\mathbf{b \leq d}$$

Habiendo hallado los vectores que describen a los esquemas relacionales y las respectivas condiciones que deben cumplir, el modelo estará dado por la siguiente definición.

Definición formal del modelo de optimización para el diseño de bases de datos relacionales.- El vector $\vec{EA}$ puede expresarse en términos de otro vector $\vec{EB}$, donde dichos vectores representan a los esquemas relacionales. Los vectores $\vec{EA}, \vec{EB} \in V^3$.

$$\boxed{\vec{EA} = s\vec{EB} + t\vec{EB}^{\perp}}$$

Con:

$$s = \frac{\vec{EA} \circ \vec{EB}}{|\vec{EB}|^2} \qquad t = \frac{\vec{EA} \circ \vec{EB}^{\perp}}{|\vec{EB}^{\perp}|^2} \quad \text{Donde } \vec{EB}, \vec{EB}^{\perp} \neq \vec{0};\ s,t \in R$$

Teniendo en cuenta que: $\vec{EA} \circ \vec{EB} = 0 \Rightarrow \vec{EA} \perp \vec{EB}; Si: \vec{EA} \neq \vec{0}; \vec{EB} \neq \vec{0}$.

Esto por el Teorema 6 descrito en el producto escalar, sirva para hallar $\vec{EB}^{\perp}$.

Entonces:

i. Cuando $\vec{EB}^{\perp}$ es perpendicular a $\vec{EA}$ y $\vec{EB}$, entonces se considera que $\vec{EB}^{\perp}$ toma los valores (1,-1,0), por lo que se tienen un traslapado de tipo completo, con la aclaración de que el término completo se refiere a que ambos vectores que representan a dos esquemas relacionales, mismos que están siendo analizados son iguales en términos de sus campos, no necesariamente en los datos, pero con la posibilidad de que si se llegue a dar.

ii. Los casos en que el vector $\vec{EB}^{\perp}$, considere los valores (1,-1,-1), este es el caso donde se tiene un traslapado de tipo relativo.

iii. Las soluciones planteadas por el modelo, se observara en el punto *f*, misma que se describe en el punto demostrativo.

Demostración.-

Demostrando el escalar s.

Partiendo de una premisa verdadera:

$$\vec{EA} = s\,\vec{EB} + t\,\vec{EB}^{\perp}$$

Multiplicando escalarmente ambos miembros por el vector $\vec{EB}$:

$$\vec{EA} \circ \vec{EB} = (s\,\vec{EB} + t\,\vec{EB}^{\perp}) \circ \vec{EB}$$

Por T3: $$\vec{EA} \circ \vec{EB} = s\,\vec{EB} \circ \vec{EB} + t\,\vec{EB}^{\perp} \circ \vec{EB}$$

Por T2: $$\vec{EA} \circ \vec{EB} = s(\vec{EB} \circ \vec{EB}) + t(\vec{EB}^{\perp} \circ \vec{EB})$$

Por T4 y T6: $$\vec{EA} \circ \vec{EB} = s\,|\vec{EB}|^2 + t(0)$$

Despejando s:

$$s = \frac{\vec{EA} \circ \vec{EB}}{|\vec{EB}|^2}$$

Por lo que queda demostrada la igualdad.

a. ***Demostrando el escalar t.***

Partiendo de una premisa verdadera:

$$\vec{EA} = s\,\vec{EB} + t\,\vec{EB}^{\perp}$$

Multiplicando escalarmente ambos miembros por el vector $\vec{EB}^{\perp}$:

$$\vec{EA} \circ \vec{EB}^{\perp} = (s\,\vec{EB} + t\,\vec{EB}^{\perp}) \circ \vec{EB}^{\perp}$$

Por T3: $$\vec{EA} \circ \vec{EB}^{\perp} = s\,\vec{EB} \circ \vec{EB} + t\,\vec{EB}^{\perp} \circ \vec{EB}^{\perp}$$

Por T2: $$\vec{EA} \circ \vec{EB}^{\perp} = s(\vec{EB} \circ \vec{EB}^{\perp}) + t(\vec{EB}^{\perp} \circ \vec{EB}^{\perp})$$

Por T4 y T6: $$\vec{EA} \circ \vec{EB}^{\perp} = s(0) + t\,|\vec{EB}^{\perp}|^2$$

Despejando t:

$$t = \frac{\vec{EA} \circ \vec{EB}^{\perp}}{|\vec{EB}^{\perp}|^2}$$

Por lo que queda demostrada la igualdad.

b. ***Demostrando la proyección completa y su significado gráfico (para este objetivo se analizara con dos vectores*** $\in V^2$ ***podrá poder visualizar de mejor manera).***

Dado dos vectores $\in V^2$ ($\vec{EA}$ = (3,2); $\vec{EB}$ = (3,2)), verificar la existencia de proyección completa.

- Hallando el valor de $\vec{EB}^{\perp}$:

Por el Teorema 6, que dice: $\vec{A} \circ \vec{B} = 0 \Rightarrow \vec{A} \perp \vec{B}; Si: \vec{A} \neq \vec{0}; \vec{B} \neq \vec{0}$

Entonces se tiene: $\vec{EB} \circ \vec{EB}^{\perp} = 0 \Rightarrow \vec{EB} \perp \vec{EB}^{\perp}; Si: \vec{EB} \neq \vec{0}; \vec{EB}^{\perp} \neq \vec{0}$

Reemplazando valores y considerando por el momento que $\vec{EB}^{\perp} = (\alpha, \beta)$

$$(3,2) \circ (\alpha, \beta) = 0$$

Resolviendo mediante el producto escalar:

$$3(\alpha) + 2(\beta) = 0$$

Ahora, para que la igualdad se de, los valores de (α, β) = (-2,3), siendo este vector perpendicular a $\vec{EB}$.

- Calculando s: (Reemplazando los valores de los vectores)

$$s = \frac{\vec{EA} \circ \vec{EB}}{|\vec{EB}|^2} = \frac{(3,2) \circ (3,2)}{3^2 + 2^2} = \frac{3(3) + 2(2)}{9 + 4} = \frac{10}{10} = 1$$

- Calculando t:

$$t = \frac{\vec{EA} \circ \vec{EB}^{\perp}}{|\vec{EB}^{\perp}|^2} = \frac{(3,2) \circ (-2,3)}{3^2 + 2^2} = \frac{3(-2) + 2(3)}{9 + 4} = \frac{-6 + 6}{10} = \frac{0}{10} = 0$$

- Reemplazando los valores hallados en $\vec{EA} = s\vec{EB} + t\vec{EB}^{\perp}$:

$$\vec{EA} = s\vec{EB} + t\vec{EB}^{\perp} = 1(3,2) + 0(-2,3) = (3,2)$$

- Representación gráfica e interpretación del ejemplo:

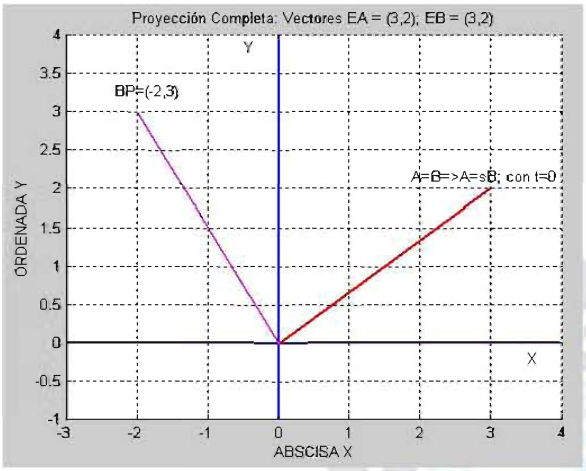


FIGURA 3.3 Ejemplo gráfico de una proyección completa

Donde: BP = $\vec{EB}^{\perp}$; A = $\vec{EA}$; B = $\vec{EB}$

Si el vector $\vec{EB}^{\perp}$ es perpendicular también al vector $\vec{EA}$, claro, considerando de que $\vec{EB}^{\perp}$ se genera a partir de $\vec{EB} \circ \vec{EB}^{\perp} = 0$, entonces se puede evidenciar que se cumple la condición i, que menciona que si dos vectores, que representan a los esquemas, ambos son perpendiculares a $\vec{EB}^{\perp}$, entonces:

$$\vec{EA} = s\vec{EB}, \quad \text{con t=0}$$

d. Demostrando la proyección relativa y su significado gráfico (para este objetivo se analizara con dos vectores $\in V^2$ ***).***

Dado dos vectores $\in V^2$ ($\vec{EA}$ = (5,3); $\vec{EB}$ = (3,3)), verificar la existencia de proyección relativa.

- Hallando el valor de $\vec{EB}^{\perp}$:

Por el Teorema 6, que dice: $\vec{A} \circ \vec{B} = 0 \Rightarrow \vec{A} \perp \vec{B}; Si: \vec{A} \neq \vec{0}; \vec{B} \neq \vec{0}$

Entonces se tiene: $\vec{EB} \circ \vec{EB}^{\perp} = 0 \Rightarrow \vec{EB} \perp \vec{EB}^{\perp}; Si: \vec{EB} \neq \vec{0}; \vec{EB}^{\perp} \neq \vec{0}$

Reemplazando valores y considerando por el momento que $\vec{EB}^{\perp} = (\alpha, \beta)$

$$(3,3) \circ (\alpha, \beta) = 0$$

Resolviendo mediante el producto escalar:

$$3(\alpha) + 3(\beta) = 0$$

Ahora, para que la igualdad se de, los valores de (α, β) = (1,-1), siendo este vector perpendicular a $\vec{EB}$.

- Calculando s: (Reemplazando los valores de los vectores)

$$s = \frac{\vec{EA} \circ \vec{EB}}{|\vec{EB}|^2} = \frac{(5,3) \circ (3,3)}{3^2 + 3^2} = \frac{5(3) + 3(3)}{9 + 9} = \frac{24}{18}$$

- Calculando t:

$$t = \frac{\vec{EA} \circ \vec{EB}^{\perp}}{|\vec{EB}^{\perp}|^2} = \frac{(5,3) \circ (1,-1)}{1^2 + (-1)^2} = \frac{5(1) + 3(-1)}{1+1} = \frac{5-3}{2} = \frac{2}{2} = 1$$

- Reemplazando los valores hallados en $\vec{EA} = s\vec{EB} + t\vec{EB}^{\perp}$:

$$\vec{EA} = s\vec{EB} + t\vec{EB}^{\perp} = \frac{24}{18}(3,3) + 1(1,-1) = (4,4) + (1,-1) = (5,3)$$

- Representación gráfica e interpretación del ejemplo:

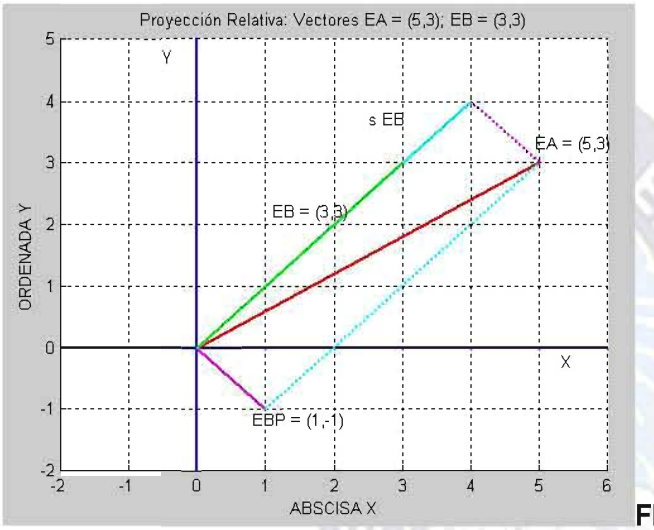


FIGURA 3.4 Ejemplo gráfico de una proyección relativa

Donde: EBP = $\vec{EB}^{\perp}$; EA= $\vec{EA}$; EB = $\vec{EB}$

Por tanto queda demostrado que cuando es una proyección relativa, verifica que ambos vectores son independientes y que existe un $\vec{EB}^{\perp}$ que no es perpendicular al vector $\vec{EA}$, por lo que se deben multiplicar los escalares s y t, como ya se hizo mención en la fórmula planteada.

e. Demostración e interpretación final del modelo de optimización para el diseño de bases de datos relacionales.

En principio, el modelo esta justificando el porque del denominativo principio de diseño ortogonal, esto en base a definiciones y teorías que se dan en los vectores.

También el modelo da una categorización del tipo de traslapado que esta siendo materia de estudio. En base al tipo de traslapado el diseñador de la base de datos puede llegar a optar por la mejor opción de diseño, para evitar este tipo de problemas.

Para demostrar que el modelo es aplicable se procederá a describir el diseño de investigaciones experimentales, mismas que se realizan con los casos descritos en el punto 3.4 de este capitulo.

Caso 1

R1 (V#, Proveedor, Status, Ciudad)
R2 (V#, Proveedor, Status, Ciudad)

- Determinando los vectores $\vec{EA}$ y $\vec{EB}$ respectivamente

Considerando las condiciones se tiene que:

$\vec{EA}$ = (4, 4, 0) $\qquad$ $\vec{EB}$ = (4, 4, 0)

$$0 \leq 0 + 0 \leq 3$$

$$4 \leq 4$$

- Hallando el vector $\vec{EB}^{\perp}$

Entonces se tiene: $\vec{EB} \circ \vec{EB}^{\perp} = 0 \Rightarrow \vec{EB} \perp \vec{EB}^{\perp} ; Si: \vec{EB} \neq \vec{0}; \vec{EB}^{\perp} \neq \vec{0}$

Reemplazando valores y considerando por el momento que $\vec{EB}^{\perp} = (\alpha, \beta, \theta)$

$$(4,4,0) \circ (\alpha, \beta, \theta) = 0$$

Resolviendo mediante el producto escalar:

$$4(\alpha) + 4(\beta) + 0(\theta) = 0$$

Ahora, para que la igualdad se de, los valores de (α, β, θ) = (1,-1,0)= $\vec{EB}^{\perp}$, siendo este vector perpendicular a $\vec{EB}$.

- Calculando s: (Reemplazando los valores de los vectores)

$$s = \frac{\vec{EA} \circ \vec{EB}}{|\vec{EB}|^2} = \frac{(4,4,0) \circ (4,4,0)}{4^2 + 4^2 + 0^2} = \frac{4(4) + 4(4) + 0}{16 + 16} = \frac{32}{32} = 1$$

- Calculando t:

$$t = \frac{\vec{EA} \circ \vec{EB}^{\perp}}{|\vec{EB}^{\perp}|^2} = \frac{(4,4,0) \circ (1,-1,0)}{1^2 + (-1)^2 + 0} = \frac{4(1) + 4(-1) + 0}{1+1} = \frac{4-4}{2} = \frac{0}{2} = 0$$

- Reemplazando los valores hallados en $\vec{EA} = s\vec{EB} + t\vec{EB}^{\perp}$:

$$\vec{EA} = s\vec{EB} + t\vec{EB}^{\perp} = 1(4,4,0) + 0(1,-1,0) = (4,4,0)$$

Siguiendo con la definición que se dio al modelo, se concluye que se tiene un traslapado de tipo completo, por ser el vector $\vec{EB}^{\perp}$ perpendicular a los vectores $\vec{EA}$ y $\vec{EB}$. Por lo que la existencia del traslapado de información en una de las tuplas es mucho más probable.

Caso 2

R1 (V#, Proveedor, Status, Cargo)
R2 (V#, Proveedor, Ciudad)

- Determinando los vectores $\vec{EA}$ y $\vec{EB}$ respectivamente

Considerando las condiciones se tiene que:

$\vec{EA}$ = (4, 2, 2) $\qquad$ $\vec{EB}$ = (3, 2, 1)

$$0 \leq 2 + 1 \leq 3$$

$$2 \leq 3$$

Hallando el vector $\vec{EB}^{\perp}$

Entonces se tiene: $\vec{EB} \circ \vec{EB}^{\perp} = 0 \Rightarrow \vec{EB} \perp \vec{EB}^{\perp}; Si: \vec{EB} \neq \vec{0}; \vec{EB}^{\perp} \neq \vec{0}$

Reemplazando valores y considerando por el momento que $\vec{EB}^{\perp} = (\alpha, \beta, \theta)$

$$(3,2,1) \circ (\alpha, \beta, \theta) = 0$$

Resolviendo mediante el producto escalar:

$$3(\alpha) + 2(\beta) + 1(\theta) = 0$$

Ahora, para que la igualdad se de, los valores de (α, β, θ) = (1,-1,-1)= $\vec{EB}^{\perp}$, siendo este vector perpendicular a $\vec{EB}$.

- Calculando s: (Reemplazando los valores de los vectores)

$$s = \frac{\vec{EA} \circ \vec{EB}}{|\vec{EB}|^2} = \frac{(4,2,2) \circ (3,2,1)}{3^2 + 2^2 + 1^2} = \frac{4(3) + 2(2) + 2(1)}{9 + 4 + 1} = \frac{18}{14} = \frac{9}{7}$$

- Calculando t:

$$t = \frac{\vec{EA} \circ \vec{EB}^{\perp}}{|\vec{EB}^{\perp}|^2} = \frac{(4,2,2) \circ (1,-1,-1)}{1^2 + (-1)^2 + (-1)^2} = \frac{4(1) + 2(-1) + 2(-1)}{1 + 1 + 1} = \frac{4 - 2 - 2}{3} = \frac{0}{2} = 0$$

- Reemplazando los valores hallados en $\vec{EA} = s\vec{EB} + t\vec{EB}^{\perp}$:

$$\vec{EA} = s\vec{EB} + t\vec{EB}^{\perp} = \frac{9}{7}(3,2,1) + 0(1,-1,-1) = (3.85, 2.57, 1.28)$$

Este resultado, según la definición se la considera como una proyección de tipo relativa. Lo que significa que cabe la posibilidad de existencia de campos con significados traslapados.

f. Posible solución planteada por el modelo

En base al estudio de los esquemas relacionales que forman parte de una base de datos, se llega a constatar que se puede llegar a dar el traslapado de información en los siguientes casos:

1. ***Entre dos entidades.*** Lo que se sugiere hacer en este caso es que se asimile mejor la semántica del problema y se redefina a todo lo que esta relacionado con el modelo conceptual. Aunque cabe señalar que este caso, en razón a la verdad, se daría en circunstancias de la poca o falta de análisis de diseño de bases de datos de parte del diseñador.

2. ***Entre dos relaciones.*** Se puede llegar a asegurar que no exista el traslapado de información mediante una buena definición de las claves primarias y foráneas. En el sistema gestor de bases de datos se lo puede realizar definiendo los CONSTRAINT, con lo que se puede llegar a formar un diagrama relacional, mismo que facilita el diseño de la base de datos.

3. ***Cuando se da en jerarquías.*** Es en este tipo donde más se puede llegar a verificar los dos tipos de traslapados ya definidos, y se dará solución en base a la definición que nos da el principio de diseño ortogonal, formulándolo de la siguiente manera:
    a. Al ser un tipo de traslapado, la varrel supertipo puede ser construido juntando las varrel subtipo.
    b. Ante la reconstrucción de la varrel supertipo, se esta asegurando de que ninguna de las proyecciones sea redundante en el proceso, por lo que se asegura la no existencia de traslapado de información.
    c. Verificar después de los cambios realizados si no ha sido afectado el proceso de normalización, específicamente cuando se afecta dos varrels con un tipo de traslapado relativo.

Capitulo 4

PRUEBA EXPERIMENTAL

El presente capitulo describe las pruebas que se realizaron para evaluar la efectividad de la solución propuesta en el capitulo 3. El punto 4.1 describe un ejemplo de un diseño de base de datos, mismo que es extraído de un texto reconocido por la comunidad informática. El apartado 4.2, se enfoca básicamente a identificar las variables e indicadores que serán objeto de prueba. La realización de los experimentos se especifica en el punto 4.3 y los resultados variando el diseño de la base de datos se lo presenta en el punto 4.4.

4.1 CONJUNTO DE DATOS UTILIZADOS

Para demostrar el modelo en la aplicación de una base de datos, se considera el planteado en el libro de Diseño de Bases de Datos (Problemas resueltos). El universo del discurso de este problema se lo mencionará textualmente, ya que en este se recoge los

requerimientos del sistema a diseñar, por lo que también se menciona los datos que se tienen que almacenar en la base de datos. El ejercicio dice lo siguiente.

GESTIÓN DE HOSPITALES

Enunciado

Una compañía aseguradora de tipo sanitario desea diseñar una base de datos para informatizar parte de su gestión hospitalaria. En una primera fase sólo quiere contemplar los siguientes supuestos semánticos:

Los hospitales de su red pueden ser propios o concertados; además de unos datos comunes a todos ellos como son el código de hospital (Cod_H), su nombre (N_H), número de camas (Num_C), etc., cuando el hospital es propio se tienen otros específicos como el presupuesto (P), tipo de servicio (TS), etc.

Una póliza, que se identifica por un número de póliza (Cod_P), tiene varios atributos que, en principio, no interesa especificar y que se agrupan bajo el nombre datos de póliza (Datos_P). Una póliza cubre a varios asegurados, los cuales se identifican por un número correlativo (Num), añadido al código de la póliza, y tienen un nombre (NA), fecha de nacimiento (FN), etc.

Los asegurados cubiertos por una misma póliza pueden ser de distintas categorías. Mientras los asegurados de primera categoría (A1C) pueden ser hospitalizados en cualquier hospital, los de segunda categoría (A2C) sólo pueden ser hospitalizados en hospitales propios. Aunque las otras categorías no tienen derecho a hospitalización, en la base de datos se guardan todos los asegurados sea cual sea su categoría.

Interesa saber en que hospitales han estado (o están) hospitalizados los asegurados, en médico que prescribió la hospitalización, así como las fechas de inicio (FI) y de fin (FF) de la misma.

Existen áreas, identificadas por un código (Cod_A) y con datos sobre su superficie (S), número de habitantes (NUM_H), etc. Los hospitales concertados tienen que estar

asignados a una única área, que no puede cambiar, mientras que los propios no están asignados a áreas.

Los médicos, que se identifican por un código (Cod_M), tienen un nombre (N_M), teléfonos de contacto, etc. Interesa conocer las áreas a las que está adscrito un médico. Existe una dependencia jerárquica entre médicos de forma que un médico tiene un único jefe.

DEFINICIÓN DEL MODELO CONCEPTUAL

Se considera el diseño de este problema por razones de que se puede observar casi todas las aplicaciones que se plantea resolver por el modelo, probando así la hipótesis.

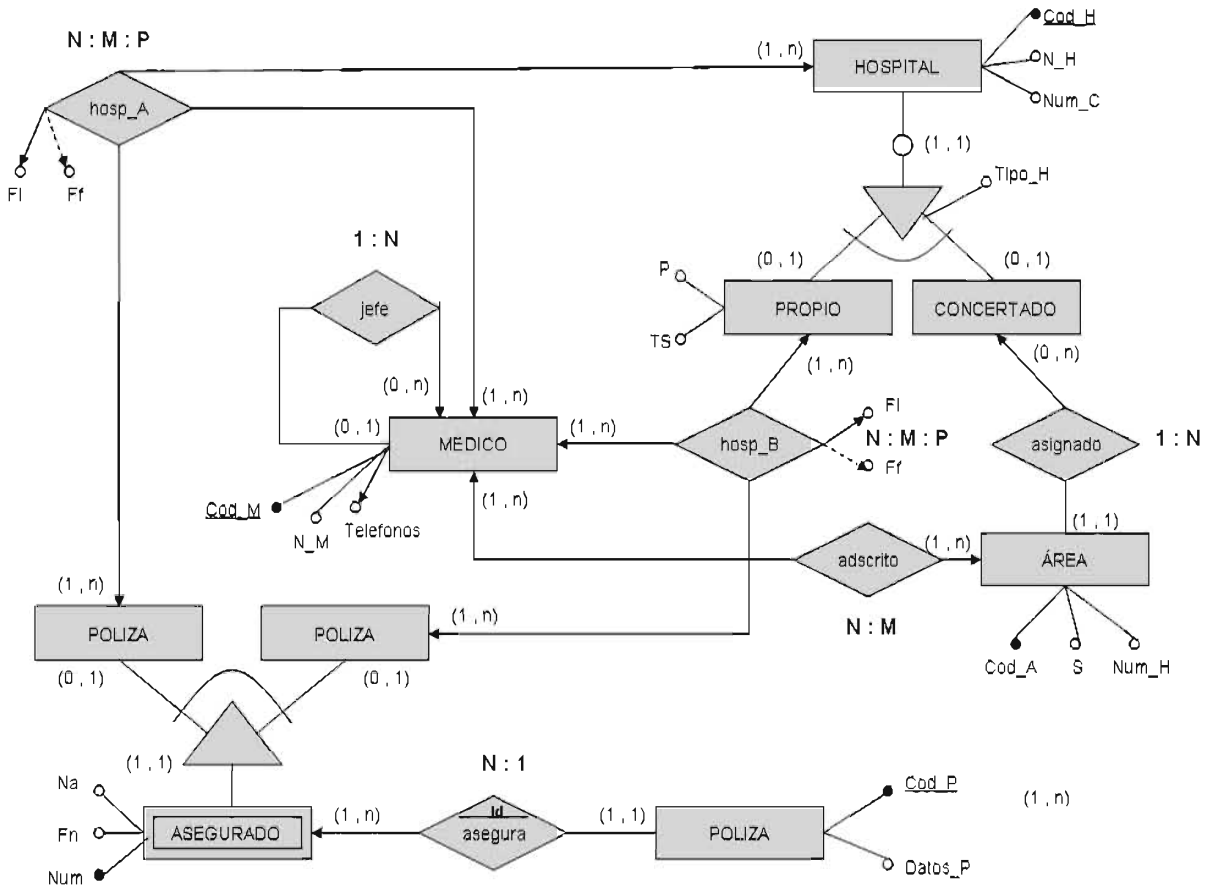


FIGURA 4.1 Diagrama E – R de la Gestión de Hospitales

4.2 VARIABLES A OBSERVAR

Las variables que se pueden llegar a identificar que llegan a tener un efecto para el comportamiento de las bases de datos, en cuanto al tema en estudio, son los siguientes.

4.2.1 Variable independiente

Los siguientes son parámetros que pueden llegar a variarse con cada muestra de datos, misma que es susceptible de ser medida dada su aplicabilidad, esto para probar una verdad tentativa. Entonces se tiene:

Incremento de variables relacionales que tienen significados que se traslapan.

Los indicadores más representativos que se identifican en base a la variable independiente son:

- *Existencia de variables relacionales con encabezados iguales o similares*. Se llega a dar en el diseño que se presenta como una muestra en las jerarquías HOSPITAL y ASEGURADO; en las relaciones HOSP_A y HOSP_B.
- *Un mal análisis del cálculo de dependencias*. La existencia de un atributo multivaluado puede llegar a causar problemas para el diseño de la base de datos. En la muestra se puede llegar a observar en la entidad MEDICO, y el atributo multivaluado teléfonos, lo que significa que existe una dependencia multivaluada.
- *La mala interpretación de requerimientos de los usuarios*. En la muestra no se llega a dar, en vista de que el enunciado que describe el diagrama E – R no existen dos entidades con los mismos atributos.
- *Mala normalización de los esquemas que conforman el sistema*.

4.2.2 Variable dependiente

La variable dependiente, se definirá para medir el efecto que se pueda tener variando la variable independiente. Entonces, se llega a definir de la siguiente manera:

Existencia de redundancia de datos.

Los indicadores que identificaron de este tipo de variable son los que a continuación se describen:

- *Mayor uso de recursos tecnológicos*.
- *Presencia de anomalías de actualización de los datos*.
- *Los datos son más complicados, en términos del entendimiento de los mismos*.

Los indicadores definidos para la variable independiente y la dependiente, se los podrán verificar en base a la métrica propuesta en el marco teórico, misma que es para medir el modelo lógico. La aplicabilidad de esta prueba se la podrá observar en el apartado de resultados.

En cuanto a los indicadores que son importantes considerar en el modelo planteado, son:

- $\vec{EB}^{\perp}$; vector perpendicular al vector $\vec{EB}$.
- $t = \dfrac{\vec{EA} \circ \vec{EB}^{\perp}}{|\vec{EB}^{\perp}|^2}$

Ambos determinan el tipo de traslapado que puede llegar a existir. Indicadores importantes para el modelo. La manipulación de estos indicadores propios del modelo se la realizara en el apartado de experimentos.

4.3 REALIZACIÓN DE LOS EXPERIMENTOS

Para los experimentos es conveniente aplicar el prototipo programado en visual Basic, el cual realiza todas las consideraciones planteadas por el modelo en su definición. Los experimentos estarán en base al modelo conceptual antes presentado, en vista de que este abarca casi todos los posibles problemas que se plantean en el modelo de optimización para el diseño lógico de las bases de datos relacionales.

Experimento 1. El traslapado de información se da entre dos relaciones

$$Hospitaliza\_A(Cod\_P, Num, Fi, Cod\_M, Cod\_H, Ff)$$

$$Hospitaliza\_B(Cod\_P, Num, Fi, Cod\_M, Cod\_H, Ff)$$

Aplicando el programa, para verificar el tipo de traslapado que se llega a dar, se tiene:

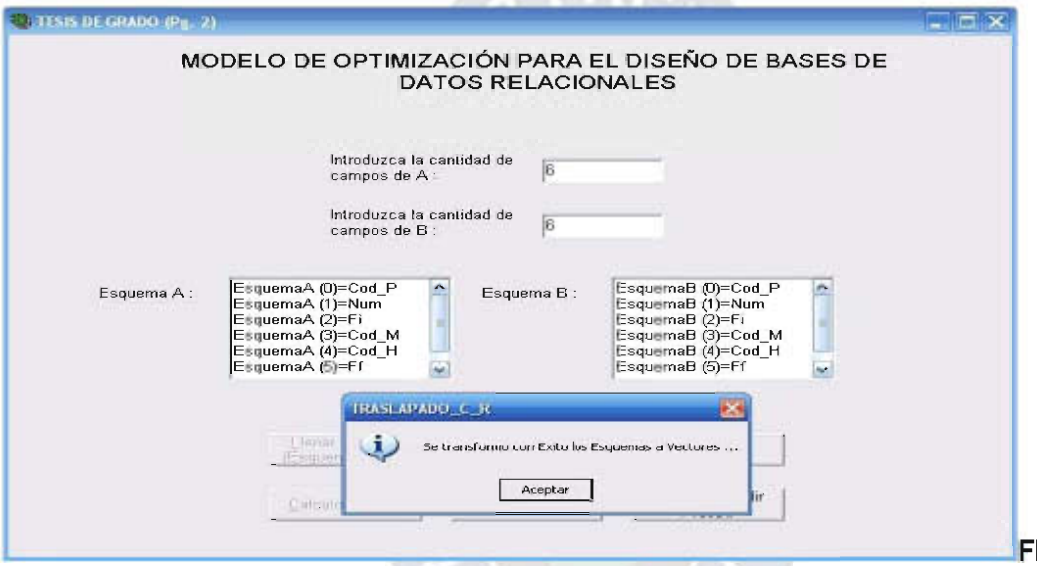


FIGURA 4.2 Experimento1 (entre dos relaciones-llenar datos)

Después de haber introducido los atributos de ambos esquemas que están siendo probados, se procede a representar a los esquemas en forma de vectores para así poder sacar las conclusiones pertinentes, entonces se tiene:

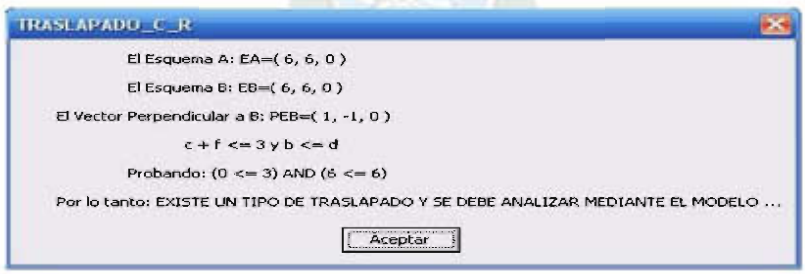


FIGURA 4.3 Experimento1 (entre dos relaciones-Verificando condiciones)

Al constatar que existe un tipo de traslapado, el modelo dice que se tiene que calcular los valores de s y t para poder así tener en concreto el tipo de traslapado, entonces:

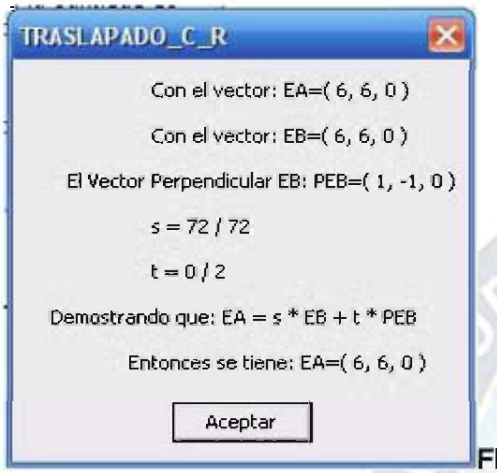


FIGURA 4.4 Experimento1 (entre dos relaciones-Tipo de Traslapado)

Al ser el vector $\vec{EB}^{\perp}$, que esta siendo representado por PEB perpendicular a $\vec{EA}$ y al mismo tiempo al vector $\vec{EB}$, se afirma entonces que se trata de un tipo de traslapado de información completo, por lo que se tienen las siguientes soluciones tentativas que da el modelo:

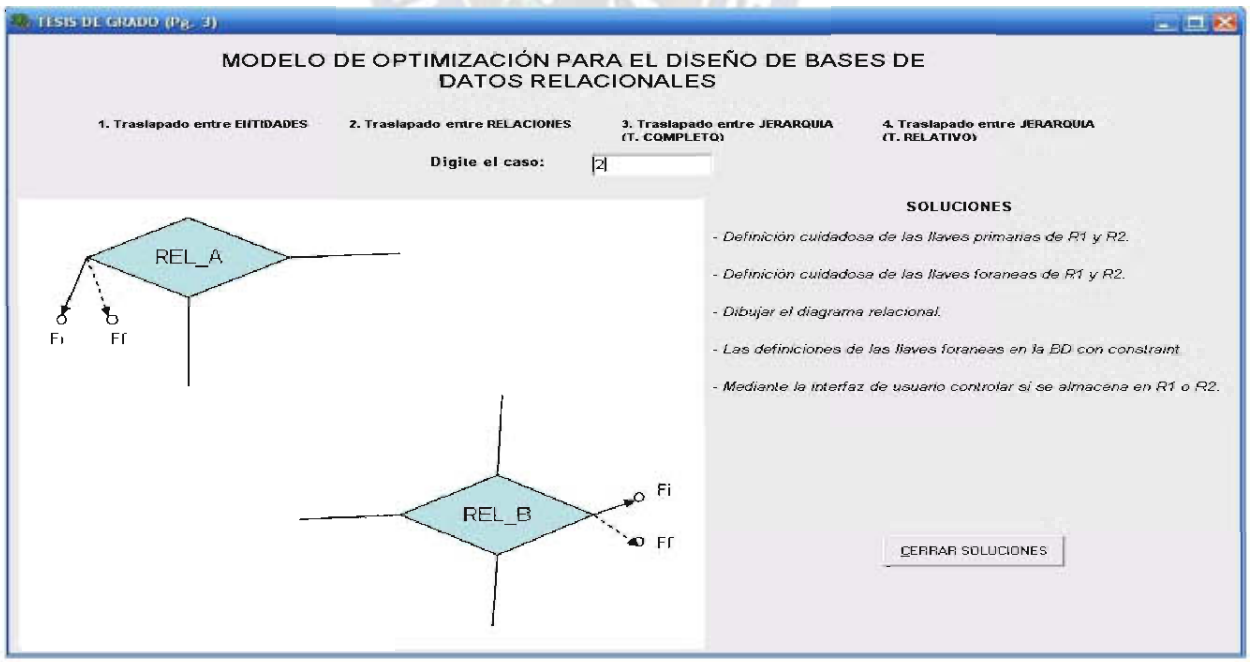


FIGURA 4.5 Experimento1 (entre dos relaciones-conclusiones)

Esta solución se la podrá verificar a detalle cuando se este realizando el diseño mismo de la base de datos, mismo que se realizara en los resultados.

Experimento 2. El traslapado en jerarquías (Relativo).

$$\Pr opio(Cod\_H, P, Ts)$$

$$Concertado(Cod\_H, Cod\_A)$$

Siguiendo con los pasos antes descritos, y aplicando el programa, que en este caso se convierte en el prototipo de prueba, se tiene:

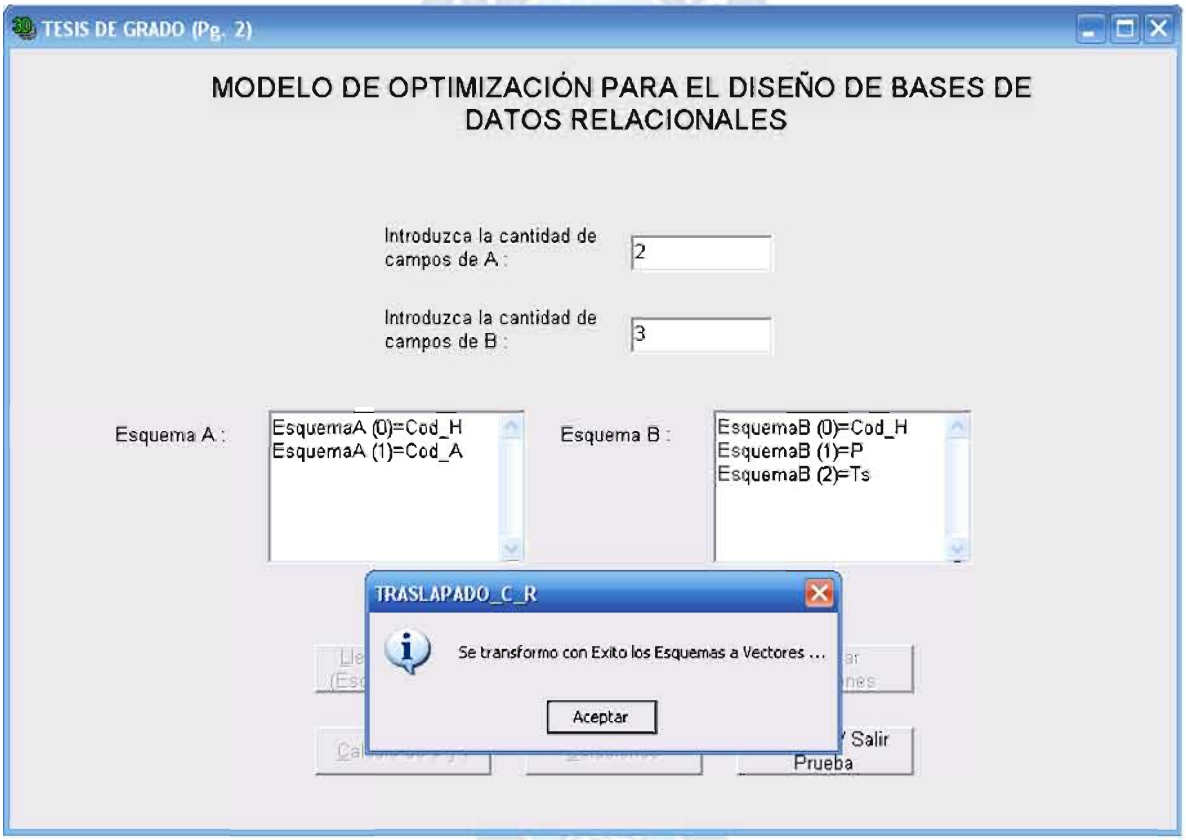


FIGURA 4.6 Experimento2 (en una jerarquía-llenar datos)

Se procede a transformar los esquemas relacionales en vectores, como en el anterior caso, entonces:

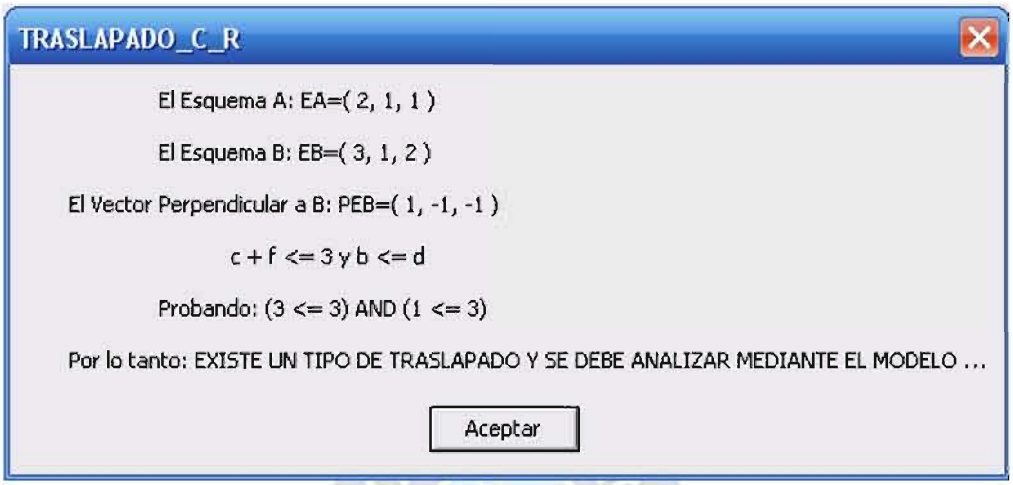


FIGURA 4.7 Experimento2 (en una jerarquía-verificando condiciones)

Ahora calculando los valores de los escalares s y t, se tiene:

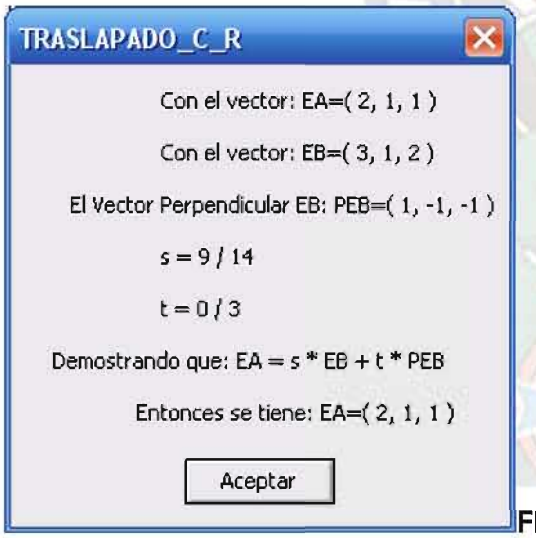


FIGURA 4.8 Experimento2 (en una jerarquía-Tipo de Traslapado)

Entonces, al verificar que $\vec{EB}^{\perp}$ es (1,-1,-1), entonces se concluye que es un traslapado de información de tipo relativo, por lo que las soluciones a este diseño es:

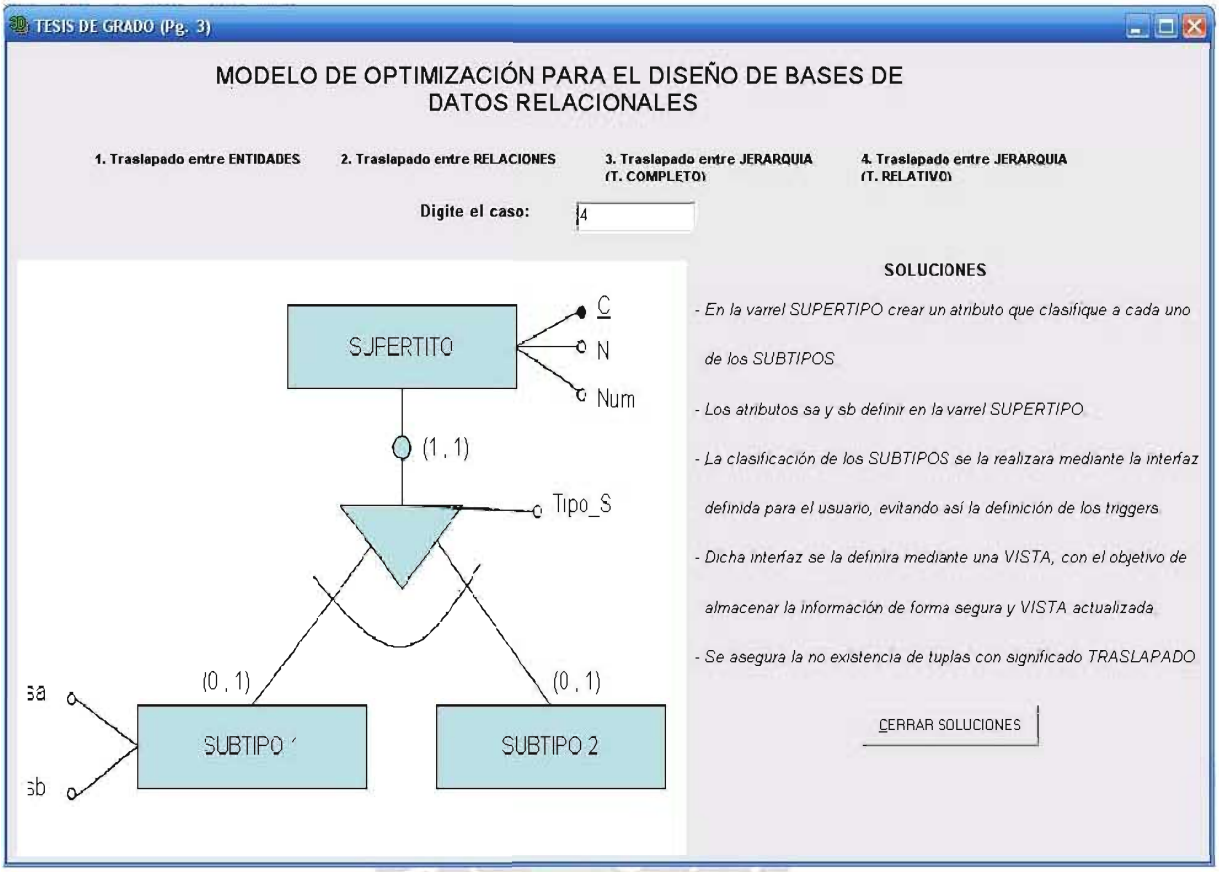


FIGURA 4.9 Experimento2 (en una jerarquía-conclusiones)

La solución se la podrá verificar mejor en el apartado de resultados.

Experimento 3. El traslapado en jerarquías (Completo)

$$A1C(Cod\_P, Num)$$

$$A2C(Cod\_P, Num)$$

El procedimiento a seguir va a ser igual que los dos anteriores, pero las consideraciones de solución cambian, por lo que considero sólo mostrar este:

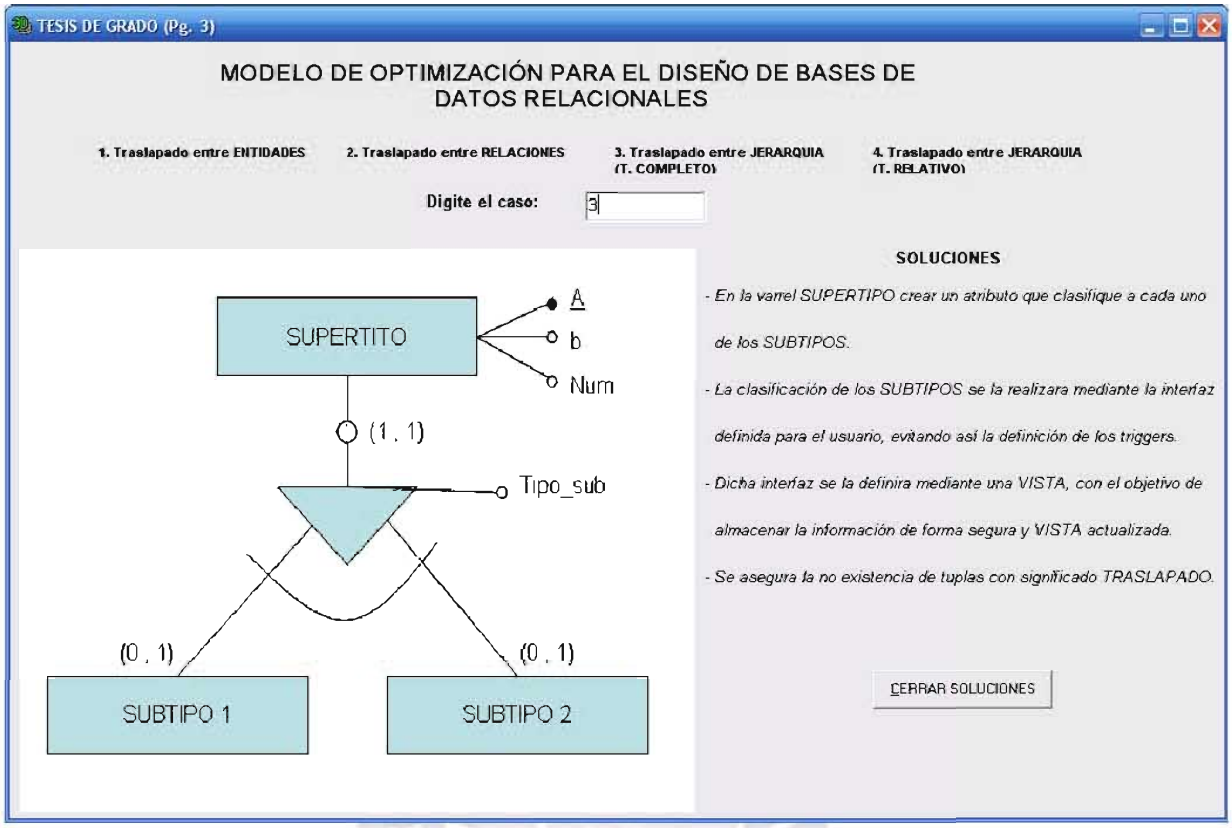


FIGURA 4.10 Experimento3 (en una jerarquía-traslapado completo)

Al ser los A1C y A2C dos esquemas subtipos, entonces se puede llegar a considerar la solución tentativa que presenta el modelo, se definirá de mejor manera en los resultados.

4.4 RESULTADOS

Para dar resultados contundentes y veraces es necesario definir al diagrama relacional que describe la figura 4.1, por lo que en primera instancia se deberá llevar al modelo relacional, esto quiere decir representarlo mediante esquemas. Cabe señalar que la normalización de los esquemas está en una Tercer forma Normal (3FN).

Con el fin de que se entienda mejor los datos que se definen en el diagrama E – R, considero importante definir los datos en términos de lo que representa en el diseño mismo, y esto se lograra definiendo el diccionario de datos.

4.4.1Diseño del modelo lógico (Gestión de hospitales)

$Hospital(Cod\_H, N\_H, Num\_C, Tipo\_H)$

$\Pr opio(Cod\_H, P, Ts)$

$Concertado(Cod\_H, Cod\_A)$

$Area(Cod\_A, S, Num\_H)$

$Medico(Cod\_M, N\_M, Jefe)$

$Telefono(Tel, Cod\_M)$

$Adscrito(Cod\_M, Cod\_A)$

$Poliza(Cod\_P, Datos\_P)$

$Asegurado(Cod\_P, Num, Na, Fn, Tipo\_A)$

$A1C(Cod\_P, Num)$

$A2C(Cod\_P, Num)$

$Hospitaliza\_A(Cod\_P, Num, Fi, Cod\_M, Cod\_H, Ff)$

$Hospitaliza\_B(Cod\_P, Num, Fi, Cod\_M, Cod\_H, Ff)$

NOMBRE ENTIDAD	DESCRIPCION
Hospital	Almacena la información sobre los hospitales que son considerados por la entidad de seguros
Propio	Almacena a los hospitales que son considerados propios de la entidad de seguros
Concertado	Almacena la información de los hospitales que son de tipo concertado
Area	Almacena la información de los datos en donde esta ubicado el hospital
Medico	Almacena los datos de los médicos
Telefono	Almacena el teléfono donde se puede recurrir a un determinado médico
Adscrito	Almacena la información donde el médico trabaja y el área
Poliza	Almacena los tipos de pólizas que se ofrecen a los asegurados
Asegurado	Almacena la información de los datos de los asegurados
A1C	Almacena los datos de los asegurados de tipo 1C, que son los que pueden ser atendidos en los hospitales propios o concertados
A2C	Almacena la información de los asegurados 2C, que sólo pueden ser atendidos en hospitales propios
Hospitaliza_A	Almacena la información de los datos de los pacientes que han sido tratados mediante la categoría A1C
Hospitaliza_B	Almacena la información de los datos de los pacientes que han sido tratados mediante la categoría A2C

FIGURA 4.11 Descripción de las entidades y relaciones

TABLA	LLAVE PRIMARIA	LLAVE FORANEA
Hospital	Cod_H	N
Propio	N	Cod_H
Concertado	N	Cod_H , Cod_A
Area	Cod_A	N
Medico	Cod_M	Jefe
Telefono	Tel	Cod_M
Adscrito	N	Cod_M , Cod_A
Poliza	Cod_P	N
Asegurado	Num	Cod_P
A1C	N	Cod_P , Num
A2C	N	Cod_P , Num
Hospitaliza_A	Fi	Cod_P , Num , Cod_M Cod_H
Hospitaliza_B	Fi	Cod_P , Num , Cod_M Cod_H

FIGURA 4.12 Especificación de tablas y llaves

Diccionario de datos

a) Tabla Hospital

NOMBRE	DESCRIPCION
Cod_H	Código del hospital
N_H	Nombre del hospital
Num_C	Número de Camas
Tipo_H	Tipo de hospital

b) Tabla Propio

NOMBRE	DESCRIPCION
Cod_H	Código del hospital
P	Presupuesto
Ts	Tipo de servicio

c) Tabla Concertado

NOMBRE	DESCRIPCION
Cod_H	Código del hospital
Cod_A	Código del área

d) Tabla Area

NOMBRE	DESCRIPCION
Cod_A	Código del área
S	Superficie del área del hospital
Num_H	Número de habitaciones

e) Tabla Medico

NOMBRE	DESCRIPCION
Cod_M	Código del médico
N_M	Nombre del médico
Jefe	Tipo de cargo de un médico

f) Tabla Telefono

NOMBRE	DESCRIPCION
Tel	Número telefónico del médico
Cod_M	Código del médico

g) Tabla Adscrito

NOMBRE	DESCRIPCION
Cod_M	Código del médico
Cod_A	Código del área

h) Tabla Poliza

NOMBRE	DESCRIPCION
Cod_P	Código de póliza
Datos_P	Descripción de la póliza

i) Tabla Asegurado

NOMBRE	DESCRIPCION
Cod_P	Código de póliza
Num	Número que cubre una póliza
Na	Nombre del asegurado
Fn	Fecha de nacimiento
Tipo_A	Tipo de asegurado

j) Tabla A1C

NOMBRE	DESCRIPCION
Cod_P	Código de póliza
Num	Número que cubre una póliza

k) Tabla A2C

NOMBRE	DESCRIPCION
Cod_P	Código de póliza
Num	Número que cubre una póliza

l) Hospitaliza_A

NOMBRE	DESCRIPCION
Cod_P	Código de póliza
Num	Número que cubre una póliza
Fi	Fecha de inicio de internación
Cod_M	Código del médico que prescribió internación
Cod_H	Código del hospital donde esta internado
Ff	Fecha de alta del paciente

m) Hospitaliza_B

NOMBRE	DESCRIPCION
Cod_P	Código de póliza
Num	Número que cubre una póliza
Fi	Fecha de inicio de internación
Cod_M	Código del médico que prescribió internación
Cod_H	Código del hospital donde esta internado
Ff	Fecha de alta del paciente

FIGURA 4.13 Tablas que definen el diccionario de datos

Ante la definición de todo lo que tiene que ver con el modelo relacional, ahora se esta en condiciones de poder entender de mejor manera el diagrama relacional que describe al diagrama E – R que se presenta en la figura 4.1, por lo que se tiene:

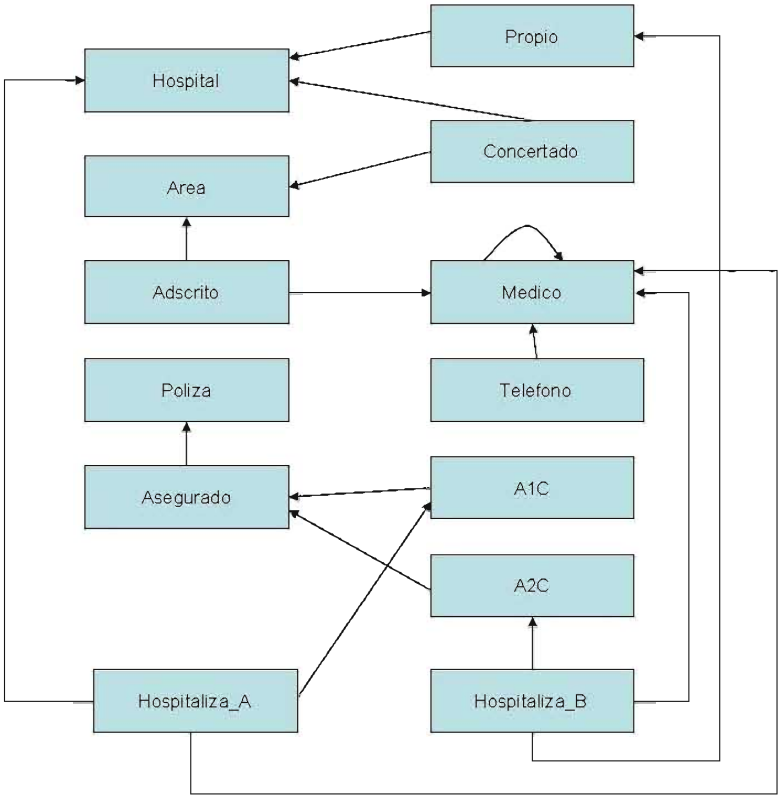


FIGURA 4.14 Diagrama referencial de la figura 4.1

Entonces, en base a las posibles soluciones que plantea el modelo y habiendo identificado los posibles esquemas relacionales donde cabe la posibilidad de que exista información traslapada, se puede llegar a formar los esquemas siguientes:

- Al existir un traslapado de tipo relativo en la jerarquía Hospital, mismo que es el supertipo de Propio y concertado, y habiendo determinado el posible traslapado de información, entonces se sigue la solución del modelo, uniendo ambos subtipos en el supertipo, lo que significa:

  $\Pr opio(Cod\_H, P, Ts)$

  $Concertado(Cod\_H, Cod\_A)$

Entonces el resultado será:

$$Hospital(Cod\_H, N\_H, Num\_C, Tipo\_H, P, Ts)$$
$$Concertado(Cod\_H, Cod\_A)$$

- En los esquemas A1C y A2C se a identificado un traslapado de tipo completo, y al ser subtipos del esquema Asegurado, siguiendo la solución tentativa se tiene que:

$$A1C(Cod\_P, Num)$$

$$A2C(Cod\_P, Num)$$

Como resultado se tendrá:

$$Asegurado(Cod\_P, Num, Na, Fn, Tipo\_A)$$

En vista de que se podrá clasificar mediante el Tipo_A, solamente.

- En los demás esquemas no existirán cambios, por lo que el diagrama relacional cambia de la siguientes manera:

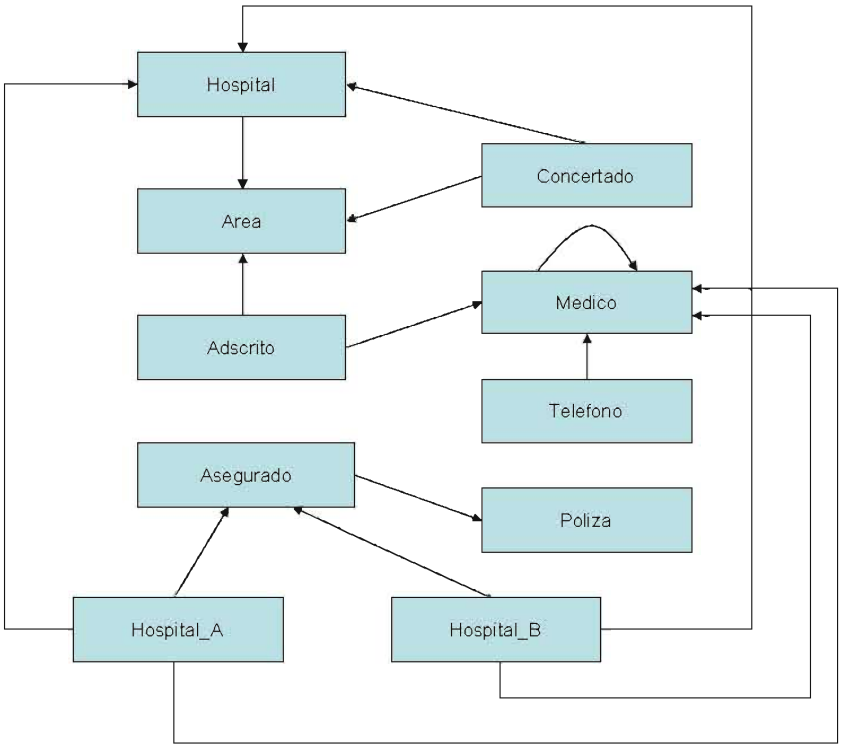


FIGURA 4.15 Diagrama referencial, aplicando soluciones del modelo planteado

Variando la variable independiente, como se puede observar la variable dependiente, que esta relacionado a la redundancia de datos tiene un decremento, con lo que se llega a probar que no existen variaciones sustanciales en los diagramas relacionales, en cuanto

se refiere a la integridad referencial, pero sí valga la redundancia a la redundancia de datos.

Una forma más concreta para demostrar este hecho es mediante la métrica presentada en el capitulo 2 para el modelo lógico.

4.4.2 Aplicación de la métrica para la evaluación de la complejidad de bases de datos relacionales

Para comprobar los beneficios de aplicar el modelo para el diseño de bases de datos es conveniente comparar los dos diagramas relacionales, esto es la figura 4.10 y la 4.11, en términos de la definición de la métrica descrita en el capitulo 2. Recordando lo que significa cada una de las variables a comparar:

NA: Número de Atributos

RD: Grado de referenciabilidad (Número de llaves foráneas (FK))

DRT: Profundidad del árbol referencial (Máximo camino referencial)

NR: Ratio de Normalidad

$NR = \frac{NT3FN}{NTS}$; NT3FN: Número de tablas en 3FN o más

NTS: Número de tablas totales

RFK: Ratio de claves ajenas de una tabla

		NA	RD	DRT	NR	RFK
1.	Hospital	4	0			0/4 = 0
2.	Propio	3	1			1/3 = 0.33
3.	Concertado	2	2			2/2 = 1
4.	Area	3	0			0/3 = 0
5.	Medico	3	1			1/3 = 0.33
6.	Telefono	2	1			1/2 = 0.5
7.	Adscrito	2	2			2/2 = 1
8.	Poliza	2	0			0/2 = 0
9.	Asegurado	5	1			1/5 = 0.2
10.	A1C	2	2			2/2 = 1
11.	A2C	2	2			2/2 = 1
12.	Hospitaliza_A	6	4			4/6 = 0.67
13.	Hospitaliza_B	6	4			4/6 = 0.67
	Totales	42	20	4	1	

FIGURA 4.16 Tabla para medir la figura 4.10

Donde:

NT: Número total de tablas = 13

Para DRT máximo:

$$Hospitaliza\_A \rightarrow A1C \rightarrow Asegurado \rightarrow Poliza$$

$$Hospitaliza\_B \rightarrow A2C \rightarrow Asegurado \rightarrow Poliza$$

Ahora bien, para el diagrama referencial propuesto, se tiene:

		NA	RD	DRT	NR	RFK
1.	Hospital	7	1			1/7 = 0.14
2.	Concertado	2	2			2/2 = 1
3.	Area	3	0			0/3 = 0
4.	Medico	3	1			1/3 = 0.33
5.	Telefono	2	1			1/2 = 0.5
6.	Adscrito	2	2			2/2 = 1
7.	Poliza	2	0			0/2 = 0
8.	Asegurado	5	1			1/5 = 0.2
9.	Hospital_A	6	4			4/6 = 0.67
10.	Hospital_B	6	4			4/6 = 0.67
	Totales	38	16	3	1	

FIGURA 4.17 Tabla para medir la figura 4.11

Donde:

NT: Número total de tablas = 10

Para DRT máximo:

$$Hospitaliza\_A \rightarrow Asegurado \rightarrow Poliza$$

$$Hospitaliza\_B \rightarrow Asegurado \rightarrow Poliza$$

Por lo que queda por demás evidente los beneficios de aplicar el modelo, ya que se puede visualizar en los resultados que se muestran en la figura 4.13 en comparación con la figura 4.12 la reducción del número de tablas, de número de atributos y del grado de referenciabilidad como los parámetros más representativos. Por lo que con estos resultados queda pues demostrada la hipótesis planteada en el capitulo 1.

Capitulo 5

CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

En el capitulo 3 se realiza la demostración de la hipótesis planteada al establecerse el modelo que optimiza el diseño de las bases de datos, con el que se soluciona el problema de la existencia de variables relacionales con significado traslapado, categorizandolos de dos tipos como los son: traslapado completo y traslapado relativo. Ahora bien, el como se plantea el modelo, diría yo de una manera poco ortodoxa en cuanto respecta al diseño de bases de datos, pero bien fundamentado en lo que es la definición de vectores y la proyección ortogonal, mismas que son base para, valga la redundancia, la definición del modelo.

La descripción del como el modelo optimiza el diseño de bases de datos con variables relacionales con significados traslapados es el siguiente:

- Analizando el diagrama relacional y haciendo una comparación de un diseño antes y después de haber aplicado el modelo, claro, dicha comparación haciendo uso de una métrica para bases de datos relacionales, se pudo probar que se puede llegar a reducir varrels base que llegarían a ser causantes de la posible existencia del traslapado de la información.
- Al identificar el tipo de traslapado de información que se puede llegar a dar, el diseñador puede llegar a tomar los recaudos necesarios para evitar este hecho, o bien considerar las posibles soluciones planteadas por el modelo.

- Con respecto a las formas normales, se ha podido constatar que ninguna aborda el traslapado de información, por lo que el tema de investigación no ha sido redundante, mas al contrario, se ha convertido en un aporte más para tener un diseño de base de datos óptimo, en términos de mantenibilidad y usabilidad.
- El contar con un diagrama relacional es de mucha ayuda, ya que así se podra visualizar de mejor manera la relación de todas las varrels. Así mismo este diagrama refleja la semántica misma del problema, por lo que es importante considerar la construcción de un diagrama relacional antes y después de aplicar el modelo para verificar que no existen anomalías de actualización de los datos.

Con relación a de que manera coadyuvaran las vistas y la tabulación cruzada, puedo concluir lo siguiente:

- Las vistas, en su definición misma son una herramienta segura para la representación de tablas a los usuarios, y el como ha sido de utilidad para el modelo, es básicamente en la introducción de datos, donde se ha podido identificar algún tipo de posibilidad de traslapado de información, además han sido de utilidad para que las mismas estén siempre actualizadas para los usuarios.
- En cuanto a la tabulación cruzada, se puede concluir que es una presentación más entendible de presentar las tablas a los usuarios, pero que se complica la construcción de la consulta misma que representa a la tabulación cruzada, además, la definición de la misma depende del tipo de sistema gestor de base de datos que se este utilizando, por lo que poco o nada ha aportado para el análisis del modelo.

Con todo lo descrito se puede afirmar que se cumplieron con todos los objetivos planteados en este trabajo de investigación.

5.2 RECOMENDACIONES

Las recomendaciones con respecto al presente trabajo son las siguientes:

- La aplicabilidad del modelo se centra en el modelo lógico, vale decir, cuando las entidades y relaciones descritas en un diagrama E – R pasan a ser varrels base, o lo que comúnmente se conoce como tablas, por lo que es recomendable que la aplicación del modelo planteado este tomado en cuenta en esta etapa.
- Se recomienda que las varrels base mínimamente estén definidas en una tercer forma normal, para que el modelo sea aplicado en forma coherente, en caso en que no se tome en cuenta esta recomendación, el modelo poco o nada puede hacer para que no exista el traslapado de información, sea está de forma completa o relativa.
- Al verificarse el tipo de traslapado completo en jerarquías, se recomienda que las varrels denominadas subtipo que son unidas a la varrel supertipo sean visualizadas como vistas, esto para no mostrar variación con el diseño original.
- En caso de que se de un tipo de traslapado relativo en jerarquías, es conveniente que se vuelva a verificar el estado de la normalización, en vista de que unir las varrels subtipo en la supertipo cabe la posibilidad de que cambie la definición de las claves candidatas, por lo que existirían dependencias funcionales que no han sido analizadas y que implicarían variaciones en la normalización de los esquemas relacionales.
- Es importante recalcar que al hacer uso de vistas en la actualización de las varrels no pueden existir valores vacios, por lo que es conveniente que a cada valor que sea introducido mediante este medio, se coloquen valores predeterminados mismos que son susceptibles a cambios.

BIBLIOGRAFIA

[ADOR00] Adoración de Miguel Castaño – Mario Piattini Velthuis

Año 2000*: Diseño de bases de datos*

México, D. F.

Editorial: ALFAOMEGA GRUPO EDITOR

[ADOP01] Adoración de Miguel Castaño – Paloma Martínez Fernández

Año 2001*: Diseño de bases de datos (Problemas Resueltos)*

México, D. F.

Editorial: ALFAOMEGA GRUPO EDITOR

[CHUQ98] Chuquimia Mamani Gladys Francisca

Año 1998*: Optimización de consultas en el diseño de una base de datos relacional*

Universidad Mayor de San Andrés

La Paz, Bolivia: Carrera de Informática

[CHUQ02] Chuquimia Poma Constancio

Año 2002*: Sistema Experto de control de normalización automática de una base de datos relacional*

Universidad Mayor de San Andrés

La Paz, Bolivia: Carrera de Informática

[COLL02] Coll M. Juan Carlos

Año 2002*: El proceso inductivo deductivo*

Disponible en <http://www.eumed.net/cursecon/ppp/induc-deduc.ppt#257,1,El proceso inductivo-deductivo>

[CHUN01] Chungara Castro Victor

Año 2001*: Apuntes y problemas de: CALCULO II*

Bolivia: Obra legalmente registrada en la Secretaria Nacional de Cultura

[DATE01] Date C. J.

Año 2001*: Introducción a los sistemas de Bases de Datos*

México: Séptima Edición

[DEGO79] De Gortari Elí

Año 1979*: El método de las ciencias. Nociones elementales*

México: Ed. Grijalbo (Tratados y Manuales Grijalbo)

[ESPI90] Espinoza Torrico Walter Saul

Año 1990*: Diseño Automático de Bases de Datos Relacionales*

Universidad Mayor de San Andrés

La Paz, Bolivia: Carrera de Informática

[FLOR06] Flores Condori Roger Diego

Año 2006*: El álgebra relacional como soporte teórico de la verificación formal de programas*

Universidad Mayor de San Andrés

La Paz, Bolivia: Carrera de Informática

[HANS00] Gary W. Hansen – James V. Hansen

Año 2000*: Diseño y Administración de Bases de Datos*

3º reimpresión

[DEEN87] S. M. Deen

Año 1987*: Fundamentos de los Sistemas de Bases de Datos*

Tercera parte: Enfoque relacional y de red (6. Modelo relacional básico)

Editorial Gustavo Gili, S. A., Barcelona 1987

[KORT02] Korth Henry F. – Silberschatz Abraham – Sudarshan S.

Año 2002*: Fundamentos de Bases de Datos*

España: Cuarta Edición

[LOPE03] López Maidana Juan Rolando

Año 2003*: Evaluación de la calidad de Bases de Datos*

Universidad Mayor de San Andrés

La Paz, Bolivia: Carrera de Informática

[MYSQ03] mysql - hispano.org

29 de Mayo del 2003*: Normalización de bases de datos*

Disponible en

<http://usuarios.lycos.es/sanjudas/download/Normalizacion%20Base%20de%20Datos.pdf>

[ORTI00] Ortiz Uribe Frida Gisela – García Nieto Ma. del Pilar

Año 2000*: Metodología de la Investigación: El proceso y sus técnicas*

México: Limusa 158 p. ilus.

[PRES03] Pressman Roger S.

Año 2003*: Ingeniería del Software*

España: Quinta Edición

[ROSA90] Rosas Lucía – Riveros Hector G.

Año 1990*: Iniciación al método científico experimental*

México: Segunda Edición

[SARD04] Lic. Sardinas Delgado Luis

Año 2004*: Investigación educativa*

España: Universidad nacional de educación de educación a distancia

(Madrid) - Comité ejecutivo de la universidad Boliviana

[TORR89] Torrico Saldaña Kattia

Año 1989*: Evaluación del desempeño de Bases de Datos Relacional*

Universidad Mayor de San Andrés

La Paz, Bolivia: Carrera de Informática

[UNAL00] Vargas Salas Carlos R.

Año 2000*: Normalización de tablas*

Laboratorio de Teledetección Aplicada y SIG – UNALM/FCF

Disponible en

<http://redinfor.lamolina.edu.pe/Separatas%20FCF/SIG%20Posicionamiento%20global/2006-I/Conceptos_Normalizacion_BD.pdf>