MÓDULO 1: ORGANIZACIÓN Y ESTRUCTURA DE LA INFORMACIÓN

Tema 3: Diseño conceptual y Normalización de datos

Leire Aldaz, Begoña Eguía y Leire Urcola

Índice del tema

Introducción
El ciclo de vida de las bases de datos
El diseño de la base de datos
El proceso de Normalización
Las Formas Normales
Reflexión final
Bibliografía

INTRODUCCIÓN

Este tema versa sobre la Normalización de datos que es una técnica que se centra en la faceta de diseño de bases de datos relacionales y que puede utilizarse para construir bases de datos ...

La etapa de diseño es fundamental en el ciclo de vida de una base de datos ya que un mal diseño acarreará problemas en las fases posteriores de interrogación y puesta al día de su información.

Si bien el estudiante del área de Ciencias Sociales y Jurídicas quizá no tenga que enfrentarse en su futura vida profesional a dicha actividad, no cabe duda de que conocer los detalles y experimentar las etapas por las que atraviesa el ciclo de la vida de una base de datos le permitirá saber sus posibilidades y limitaciones, las preguntas que se le pueden formular y las respuestas que razonablemente se pueden extraer de ella y, además, este conocimiento le ayudará a abordar un nuevo diseño o el mantenimiento de un esquema de base de datos que ya exista en la empresa u organización en la que trabaje.

El Ciclo de vida de la base de datos

La base de datos es uno de los componentes principales de un sistema de información, pudiendo convertirse en su verdadera columna vertebral, por lo que se puede afirmar que el ciclo de vida del sistema de información está inherentemente ligado al ciclo de vida de la base de datos sobre la que se apoya.

Las etapas de este ciclo de vida son básicamente: planificación de la base de datos, definición del sistema, recolección y análisis de requisitos, diseño de la base de datos, selección del sistema de gestión de bases de datos apropiado, diseño de aplicaciones, elaboración de prototipos, implementación, conversión y carga de datos, prueba y mantenimiento.

Desde una visión general, el ciclo de vida de una base de datos puede descomponerse en tres etapas, que son: concepción, creación de la base de datos y su explotación. La fase de concepción o diseño consiste en reproducir el mundo real con ayuda de uno de los modelos de datos[1], ya sea: jerárquico, en red, relacional, orientado a objetos u otro distinto que ayude a entender el significado de los datos y que facilite la comunicación en cuanto a los requisitos de información. (Basándonos en el modelo relacional, se trata de traducir el universo real en términos de tablas. Estas tablas representan entidades del mundo real así como los enlaces que pueden existir entre ellas).

La segunda fase consiste en traducir el esquema conceptual en órdenes comprensibles por el Sistema de Gestión de Bases de Datos. Hay que recordar que la base de datos contiene los datos y su definición pero todos los accesos a la misma se realiza a través de un interface común que proporciona el sistema de gestión de la base de datos. Básicamente el SGBD es un conjunto de programas que permite la creación, consulta y manipulación de los datos a través de los lenguajes que proporciona, una para la definición (LDD) y el otro para la manipulación de los mismos.

[1] Un modelo de datos conceptual consta de entidades, interrelaciones, atributos, etc.

Por último se lleva a cabo la manipulación de datos, los usuarios pueden consultar los datos y ponerlos a punto, mediante las operaciones de actualización correspondientes durante todo el resto de la vida de la base de datos.

La capacidad de un modelo de datos de responder a las cuestiones que se le plantean queda claramente determinada primero, por su compleción (obviamente, ningún sistema de BD puede proporcionar datos de los que no disponga) y segundo por su estructura. No obstante, la facilidad con la que se puedan responder las preguntas depende casi exclusivamente de este segundo aspecto.

Realizar un buen diseño es crucial para el resto del ciclo de vida de la base de datos, ya que al igual que en toda etapa de concepción, una mala concepción de la BD entrañará problemas en el futuro. De ahí que se insista en esta etapa.

La etapa del Diseño de la base de datos

Al hablar del diseño hay que distinguir básicamente tres tipos: el diseño conceptual, el diseño lógico y el diseño físico.

El diseño conceptual consiste en construir un esquema global de la información que maneja la empresa, independientemente de todas las consideraciones físicas.

En el diseño lógico el esquema anterior se transforma según el modelo lógico que se vaya a utilizar para implementar la base de datos, en nuestro caso, el modelo lógico relacional.

Por último, en la etapa del diseño físico, se produce el esquema interno o una descripción detallada de la implementación de la base de datos en el ordenador.

El objetivo principal de la etapa del diseño lógico es sencillo: asegurar que el modelo que se ha diseñado sea capaz de responder a cualquier pregunta que, razonablemente, se pueda plantear sobre él.

Existen dos enfoque o formas de proceder distintas:

a) El primer enfoque asume la creación de un modelo de datos conceptual[2] durante el diseño conceptual. Este modelo es convertido mecánicamente a un modelo relacional. Para ello, se deben hacer algunas transformaciones en él: como eliminar las asociaciones de muchos a muchos, eliminar las relaciones complejas, eliminar las relaciones recursivas, eliminar los atributos multievaludados, etc. El proceso de conversión garantizará la validez de los resultados obtenidos del modelo conceptual (obteniéndose relaciones normalizadas para la cuarta forma normal).

b) El segundo enfoque es más tradicional. En este enfoque no se parte del modelo de datos conceptual, sino que se procede directamente a la creación de un esquema relacional aproximado a partir de la observación directa del mundo real. El diseño se completa mediante la **normalización** de las definiciones de las tablas que han sido generadas.

En ambos casos se aplica la normalización, sin embargo en el primer enfoque se aplica como técnica de validación para garantizar que el diseño resultante no viole de manera no intencional cualquiera de los principios de la normalización.

[2] Hay varias formas de representar el esquema conceptual, pero la mas utilizada es el modelo Entidad/Relación de (Chen, 1976), en el que la estructura de datos se representa mediante Entidades, Atributos, Relaciones... (una relación es una asociación no trivial entre los componentes o entidades de un modelo, cuya no inclusión supone una pérdida de información. El grado o correspondencia de las relaciones puede ser 1 a 1, 1 a n, n a m. Las relaciones n a m son muy frecuentes en la práctica real. Ejemplo: autor escribe libros. Pero estas asociaciones no se pueden implementar directamente en el modelo relacional, hay que descomponerlas.

En el segundo enfoque la normalización es un paso obligado cuando el esquema relacional se obtiene directamente desde la observación del mundo real, sin pasar por la construcción de un modelo conceptual. Nosotros escogemos este segundo enfoque, adoptando la suposición útil de que en realidad se está llevando a cabo el proceso de diseño mediante la aplicación directa de este procedimiento ya que proporciona una infraestructura conveniente a partir de la cual podemos enunciar los principios subyacentes.

Comparativamente hay que señalar que el primer enfoque, en el que se utilizan modelos conceptuales, es válido principalmente en el diseño de grandes y complejos esquemas de bases de datos corporativas, mientras que el segundo, está orientado fundamentalmente, a situaciones que requieren un esquema de bases de datos relativamente simple.

¿Pero qué es el proceso de Normalización, en qué consiste?

El proceso de normalización

La normalización es una técnica para diseñar la estructura lógica de los datos de un sistema de información en el modelo relacional, fue desarrollada por E. F. Codd en el año 1972. Es una metodología de diseño *bottom up*; donde se parte de una serie de atributos, propiedades o características de los datos, y éstos se van agrupando en relaciones o tablas según su afinidad.

La idea es sencilla: un conjunto dado de relaciones (tablas) es reemplazado por otro conjunto de tablas con una estructura más simple y más regular con el objetivo principal de minimizar en lo posible la redundancia de información y evitar ciertas anomalías que pueden producirse en el mantenimiento de la base de datos.

Entre las ventajas principales de la normalización cabe destacar las siguientes:

- Evita anomalías en inserciones, modificaciones y borrados.
- Mejora la independencia de datos.
- No establece restricciones artificiales en la estructura de los datos.

Para llevar a cabo el proceso de normalización existen varias formas normales. Las tres primeras formas normales estaban contempladas en la formulación original de la teoría relacional de Codd.. Más adelante, Boyce y Codd definieron una versión modificada de la tercera forma normal, y posteriormente fueron definidas la cuarta y la quinta forma normal originando un mayor refinamiento en el diseño de bases de datos.

No obstante continuar más allá de la forma normal de Boyce Codd tiene un dudoso valor práctico, ya que las siguientes formas normales se desarrollaron para manejar casos especiales, siendo la mayoría de ellos bastante raros.

Fases de la normalización

Cada paso de la normalización consiste en reducir sistemáticamente una relación o tabla en una colección de tablas "más pequeñas" (es decir, de menor grado) que son equivalentes a la de partida. Es importante señalar que este procedimiento es reversible, es decir, siempre es posible tomar los resultados del procedimiento y transformarlos de nuevo a su origen. Esta característica es importante, ya que significa que el proceso de normalización conserva la información.

El proceso general puede enunciarse de manera informal como un conjunto de fases, de la siguiente manera:

Inicialmente, como resultado de la fase de observación del mundo real, se construye un esquema relacional aproximado, en el que se representan las tablas y sus atributos.

La técnica más utilizada es la que consiste en construir una sola tabla, llamada tabla Universal, que engloba a todos los atributos que se van a manejar.

Cada paso de la normalización corresponde a una forma normal que satisface unas reglas o propiedades determinadas y, conforme se va avanzando en este proceso las nuevas relaciones o tablas que se van generando tienen un formato más estricto (más fuerte) y, por lo tanto, son menos vulnerables a los comportamientos anómalos que puedan darse en las operaciones de actualización de datos.

Veamos cómo se aplica la técnica de normalización a un caso práctico.

Caso práctico

Supongamos que un inexperto en bases de datos tiene que construir una base de datos para una pequeña empresa que se dedica a la venta de artículos. Tras preguntar a personal de la empresa éstos le explican el funcionamiento general de la misma indicándole las consideraciones que tiene que tener en cuenta.

- Los clientes realizan pedidos del o de los artículos que desean.
- Cada vendedor, empleado de la empresa, recoge pedidos de los clientes.
- Cada pedido puede estar formado por uno o varios artículos.
- Existen dos precios para cada artículo, el precio de venta al público y el precio de venta recomendado (al cliente)
- Una vez confeccionado el pedido se le comunica al cliente y si éste lo acepta se emite la correspondiente factura, en caso contrario se anula la operación.

El primer paso consiste en recogen los datos que hay que manejar:

	DATOS DEL CLIENTE	DATOS DEL PEDIDO	DATOS DEL ARTÍCULO	DATOS DE FACTURA	DATOS DEL EMPLEADO
ATRIBUTOS	CLI#	PE#	ART#	FACT#	EMPL#
	Nombre	Fecha	Descripción	Fecha	Nombre
	Cif	Total	Existencias	Importe	Categoría
	Direcci		PVP		
	Ciudad		PV-CLI		
	Cod-Postal		Cant		
	Teléfono				

Supongamos que se decide diseñar un esquema sencillo formado por una sola Tabla. La tabla correspondiente a dicha entidad sería la siguiente:

PEDIDO (CLI#, Nombre, CIF, Direcc, Ciudad, PE#, Fecha, Total,
(ART#, Desc, Existencias, PVP, PV-Cli, Cant),
FACT#, Fecha, Importe, EMPL#, Nombre, Categoría**)**

Una posible extensión de esta tabla (Para facilitar la lectura de la tabla se ha considerado oportuno no presentar la totalidad de los datos a manejar, sino tan sólo un subconjunto de los mismos):

Extensión de la tabla:

CLI#	Razon	CIF	Direcc	Ciudad	PE#	ART#	Desc	Q	Fecha
1250	Hnos. López	B9517 7051	Marina, nº 41	Vigo	1020	1035 2241 2518 1090 1053	Regrabador DVD Impresora i45 Escaner, L30 Palm, Sony-25 Flash 256 Mb	5 2 2 3 6	29/11 /03
1250	Martín	F20033 361	Ayala, nº 14	Toledo	1035	2241	Impresora i45	5	30/11 /03
1251	Ramos	A2037 2306	Mayor, nº 6	Segovia	1040	1035 2241	Regrabador DVD Impresora i45	4 9	02/12 /03
1251	Ramos	A2037 2306	Mayor, nº 6	Segovia	1045	2518	Escaner, L30	5	03/12 /03

Clave

Universidad del País Vasco · Euskal Herriko Unibertsitatea

Además, es conveniente definir una clave, es decir, hay que buscar entre todos los atributos aquel o aquellos cuyo valor puedan servir para identificar de forma directa y por lo tanto rápida a un elemento concreto. Puede haber varios, que cumplan este requerimiento, pero solo uno será definido como Clave principal el resto, quedarán como claves alternativas.

Entre todos los atributos existentes, el Número del Pedido puede definirse como Clave principal.

Si todos estos datos se engloban en única tabla, llamada tabla universal, se producirán los siguientes problemas o anomalías:

- **Anomalías de repetición**: puede que cierta información esté repetida innecesariamente.
- **Anomalías de modificación**: como consecuencia de las repeticiones, las modificaciones o cambios a realizar pueden afectar a múltiples filas.
- **Anomalías de inserción**: quizás puede resultar imposible añadir información a la BD.
- **Anomalías de borrado**: el borrado de una fila podría implicar pérdida de información relevante.

La redundancia no es buena, no sólo porque se desperdician recursos, sino también porque nos complica la vida. Pongamos como ejemplo el conjunto de registros que se muestra en la diapositiva, que representa parte de la información de los pedidos que llegan a la empresa.

Como se puede observar se repiten tantas veces los datos generales asociados a un cliente, como su nombre o razón social, nif, dirección, ciudad como pedidos haya realizado éste cliente; mas concretamente, como artículos distintos haya solicitado en cada uno de sus pedidos.

De la misma manera, aunque no pueda apreciarse en la tabla presentada, se repetiría la información general asociada a los artículos, tantas veces como sean solicitados en distintos.

Estas redundancias tienen algunas consecuencias. En primer lugar, significa que cada vez que llega un pedido hay que introducir de nuevo en el sistema los valores en esos campos. Y, por su puesto, cada vez que se introduce algo se corre el riesgo de equivocarse. Luego los datos duplicados pueden llevar a inconsistencias. Nos podríamos encontrar con direcciones distintas para un mismo cliente o descripciones distintas para un mismo artículo.

En segundo lugar, esta estructura de datos hace que quizá no se pueda añadir información de un cliente nuevo que la empresa haya captado hasta que dicho cliente no efectúe algún pedido. Esto es debido a que el valor de la clave primaria no puede ser Nulo, es una de las restricciones de la integridad de entidad del modelo Relacional, que hay que tener presente. Concretamente este tipo de Integridad o restricción debe asegurar que las claves primarias no admitan nulos.

De la misma manera, si un artículo nuevo llega a almacén sus datos generales no podrán introducirse mientras dicho artículo no sea solicitado.

Por último, al eliminar un pedido cabe la posibilidad de que se pierda toda la información asociada a un cliente y/o a un artículo, caso de que el cliente hubiese realizado un único pedido de un artículo no solicitado otras.

Estos problemas, a menudo denominados *anomalías de actualización*, se acentúan si los datos redundantes se guardan en más de una estructura de datos. Para evitar estas anomalías o comportamientos anómalos se propone el proceso de Normalización que se lleva a cabo a través de la aplicación de las reglas de las distintas formas normales.

Pero antes de pasar a enunciarlas es necesario estar familiarizado con una serie de principios subyacentes. Estos principios son básicamente herramientas

para controlar la estructura de los datos (de la BD): la descomposición sin pérdida y la dependencia funcional.

Descomposición sin pérdida

El método más usado de eliminar la redundancia consiste en descomponer o dividir iterativamente una relación en dos o mas relaciones, de tal forma que las relaciones resultantes se puedan combinar sin que se pierda ninguna información. Este es el principio de la descomposición sin pérdida.

El modelo relacional permite que las relaciones se unan de varias formas vinculando atributos.

Ejemplo: ¿Qué pedidos ha realizado el cliente Ramos?

CLIENTES

CLI#	Nombre	CIF	Direcc	Ciudad
12500	Hnos. López	15.123.567-F	Marina, 41	Vigo
12505	Martín	34.009.876-M	Ayala, 14	Toledo
12510	Ramos	15.456.768-N	Mayor, 6	Segovia

PEDIDO

PE#	Fecha	CLI#	Total
1020	29/01/03	12500	
1035	30/01/03	12505	
1040	02/02/03	12510	
1045	03/02/03	12510	

Dependencia funcional

La normalización se basa en la Teoría de la dependencia funcional. (Uno de los conceptos fundamentales en la normalización es el de ***dependencia funcional***.) La dependencia funcional se da (produce) entre atributos de una misma relación (tabla).

Si ***x*** e ***y*** son atributos de la relación ***R***, se dice que ***y*** es funcionalmente dependiente de ***x*** (se denota por ***x*** ➔ ***y***) si cada valor de ***x*** tiene asociado un solo valor de ***y*** sin ambigüedad. A ***x*** se le denomina determinante, ya que ***x*** determina el valor de ***y***. Se dice que el atributo ***y*** es completamente dependiente de ***x*** si depende funcionalmente de ***x*** y no depende de ningún subconjunto de ***x***.

La dependencia funcional es una noción semántica. Si hay o no dependencias funcionales entre atributos no lo determina una serie abstracta de reglas, sino, más bien, los modelos mentales del usuario y las reglas de negocio de la organización o empresa para la que se desarrolla el sistema de información.

Así, el reconocimiento de las dependencias funcionales es parte del proceso de entender lo que significan los datos (en una situación determinada).

En la práctica, la dependencia funcional es una forma conveniente de expresar un concepto que resulta bastante evidente: dada una relación, debe haber algún conjunto de atributos que resulte único para cada tupla o elemento de la tabla, y, sabiendo eso se pueden determinar los atributos que no son únicos.

En nuestro ejemplo:

IdPedido ➔IdCliente
IdCliente ➔Nombre o Razón Social, Dirección, ...

Para un mismo identificador de cliente se tendrá siempre el mismo valor en los atributos Razón social, dirección, etc.

(Hay que tener en cuenta que la relación inversa no tiene porqué cumplirse necesariamente. Es decir, hay que contemplar la posibilidad de que clientes de la misma empresa puedan efectuar pedidos, luego conociendo únicamente el nombre del cliente o la razón social, no se puede extraer directamente el cliente que ha realizado el pedido (ya que puede haber varios).

De la misma manera:

IdArticulo ➔ Descripción, Existencias…

Pero ¿conociendo únicamente la identificación de un artículo puede determinarse qué cantidad se ha pedido? IdArticulo ➔ ¿Cantidad? No. Dependerá también del pedido en el que se haya solicitado.

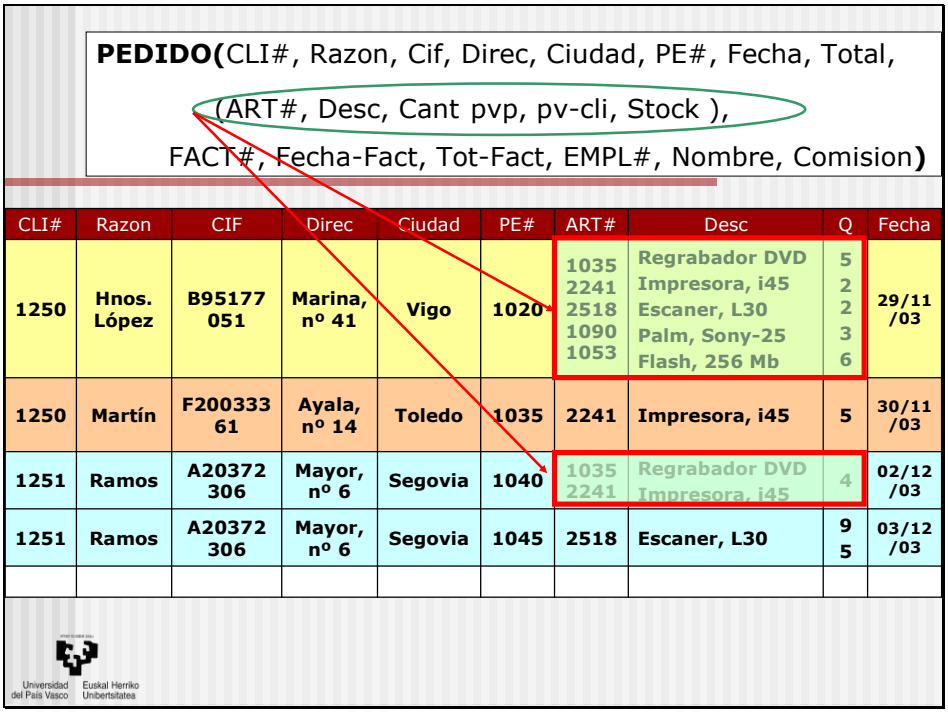
Una vez aclarados estos principios básicos, a continuación se enunciarán de manera informal las reglas que tienen que satisfacerse para que una estructura relacional se encuentre en una determinada forma normal. Hay que señalar que si bien la terminología empleada es un tanto esotérica las ideas que se exponen son esencialmente muy sencillas y de sentido común. De cualquier forma, algunas referencias que se presentan al final de la lección tratan este material de una manera mucho más formal y rigurosa.

Primera Forma Normal (1FN)

Una relación está en primera forma normal si, y sólo si, todos los dominios de la misma contienen valores atómicos, es decir, no hay grupos repetitivos o multievaluados. Si se ve la relación gráficamente como una tabla, estará en 1FN si tiene un solo valor en la intersección de cada fila con cada columna (cada registro contiene exactamente un valor para cada atributo).

Luego nuestra tabla no se encuentra en 1FN, para pasar a 1FN hay que eliminar de ella los grupos repetitivos. Un grupo repetitivo será el atributo o grupo

de atributos que tienen múltiples valores para cada registro de la relación. Hay dos formas de eliminar los grupos repetitivos.



CLI#	Razon	CIF	Direc	Ciudad	PE#	ART#	Desc	Q	Fecha
1250	Hnos. López	B95177 051	Marina, nº 41	Vigo	1020	1035 2241 2518 1090 1053	Regrabador DVD Impresora, i45 Escaner, L30 Palm, Sony-25 Flash, 256 Mb	5 2 2 3 6	29/11 /03
1250	Martín	F200333 61	Ayala, nº 14	Toledo	1035	2241	Impresora, i45	5	30/11 /03
1251	Ramos	A20372 306	Mayor, nº 6	Segovia	1040	1035 2241	Regrabador DVD Impresora, i45	4	02/12 /03
1251	Ramos	A20372 306	Mayor, nº 6	Segovia	1045	2518	Escaner, L30	9 5	03/12 /03

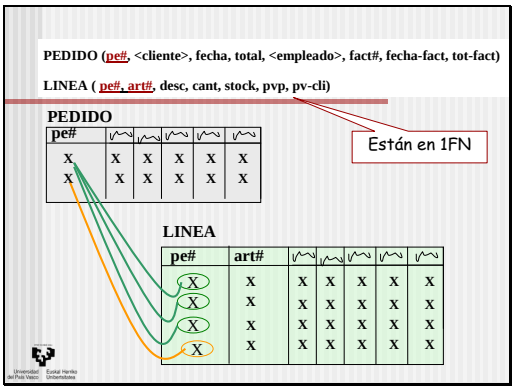
a) En la primera, se repiten los atributos con un solo valor para cada valor del grupo repetitivo. De este modo, se introducen redundancias ya que se duplican valores, pero estas redundancias se eliminarán después mediante las restantes formas normales.

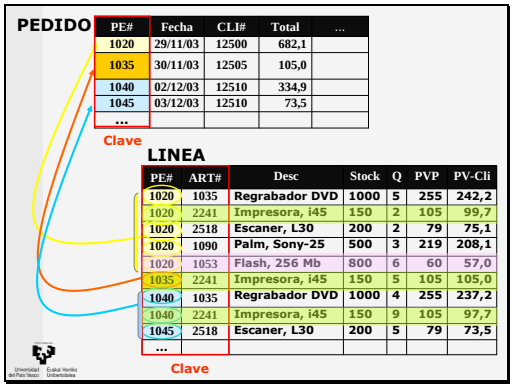
CLI#	Razon	Cif	Direc	Ciudad	PE#	ART#	Desc	Q	FECHA
12500	Hnos. López	B9517 7051	Marina, 41	Vigo	1020	1035	Regrabador DVD	5	29/11 /03
12500	Hnos. López	B9517 7051	Marina, 41	Vigo	1020	2241	Impresora i45	2	29/11 /03
12500	Hnos. López	B9517 7051	Marina, 41	Vigo	1020	2518	Escaner, L30	2	29/11 /03
12500	Hnos. López	B9517 7051	Marina, 41	Vigo	1020	1090	Palm, Sony-25	3	29/11 /03
12500	Hnos. López	B9517 7051	Marina, 41	Vigo	1020	1053	Flash, 256 Mb	6	29/11 /03
12505	Martín	F20033 361	Ayala, 14	Toledo	1035	2241	Impresora i45	5	30/11 /03
12510	Ramos	A2037 2306	Mayor, 6	Segovia	1040	1035	Regrabador DVD	4	02/12 /03
12510	Ramos	A2037 2306	Mayor, 6	Segovia	1040	2241	Impresora i45	9	02/12 /03
12510	Ramos	A2037 2306	Mayor, 6	Segovia	1045	2518	Escaner, L30	5	03/12 /03

Redundancias

Clave principal: **IdPedido+IdArticulo**

b) La segunda forma de eliminar los grupos repetitivos consiste en poner cada uno de ellos en una relación (tabla) aparte, heredando la clave primaria de la relación en la que se encontraban. Esta es la alternativa que vamos a tomar.





Segunda Forma Normal (2FN)

Una relación está en 2FN si, y sólo si, está en 1FN y, además, cada atributo que no es clave primaria depende completamente de la totalidad de la clave y no de un subconjunto de ella.

Así, la 2FN se aplica así a las relaciones que tienen claves primarias compuestas por dos o más atributos. Si una relación está en 1FN y su clave primaria es simple (formada por un solo atributo), entonces también está en 2FN. Las relaciones que no están en 2FN pueden sufrir anomalías cuando se realizan actualizaciones luego es conveniente descomponerlas

Descomposición.

Para pasar una relación de 1FN a 2FN hay que eliminar las dependencias parciales de la clave primaria. Para ello, se pasan los atributos que son funcionalmente dependientes a una nueva relación (tabla) con una copia de su determinante, y pasando a ser éste clave primaria de la nueva relación generada.

Llegados a este punto, nos tenemos que cuestionar: ¿sería conveniente seguir con el proceso de descomposición (seguir descomponiendo aún más)?

Si bien las relaciones en 2FN tienen menos redundancias que las relaciones en 1FN, todavía pueden sufrir anomalías frente a las actualizaciones. Luego vamos a seguir avanzando en el proceso de descomposición…

Tercera Forma Normal (3FN)

Una relación está en 3FN si, y sólo si, ya se encuentra en 2FN y, además, cada atributo que no es clave primaria no depende transitivamente de la clave primaria. La dependencia $x \rightarrow z$ es transitiva si existen dependencias $x_{clave} \rightarrow y$, $y_{no\ clave} \rightarrow z$, siendo x, y, atributos o conjuntos de atributos de una misma relación

(En nuesto ejempo: IdPedido➔IdCliente,
Pero particularmente IdCliente➔Razon-Social, Direcc, Telef…)

Descomposición:

Para pasar una relación de 2FN a 3FN hay que eliminar las dependencias transitivas. Para ello, se eliminan los atributos que dependen transitivamente y se ponen en una nueva relación con una copia de su determinante (el atributo o atributos no clave de los que dependen) pasando a ser estos clave primaria en la nueva relación.

Forma Normal de Boyce-Codd (BCFN)[3]

Una relación está en la forma normal de Boyce-Codd si, y sólo si, todo determinante es una clave candidata (definición informal).

La 2FN y la 3FN eliminan las dependencias parciales y las dependencias transitivas de la clave primaria. Pero este tipo de dependencias todavía pueden existir sobre otras claves candidatas, si éstas existen.

La violación de la BCFN es poco frecuente ya que se da bajo ciertas condiciones que raramente se presentan.

Es recomendable comprobar si una relación viola la BCFN en el caso que tenga dos o más claves candidatas compuestas por al menos un atributo común.

De hecho, nuestro esquema relacional ya se encuentra en la BCFN. La normalización puede continuar, dando lugar a un mayor refinamiento en el diseño de una base de datos, mediante la aplicación de más forma normales (4FN, 5FN; etc.). Pero el interés teórico de este proceso, continuar más allá de la forma normal de Boyce-Codd tiene un dudoso valor práctico, ya que la mayoría de los productos comerciales no soportan formas normales superiores a esta última.

No obstante, vamos a definir también la 4FN para el caso en que fuera necesaria su aplicación.

[3] La BCFN es más fuerte que la 3FN, por lo tanto, toda relación en BCFN está en 3FN.

Cuarta Forma Normal (4FN)

Se dice que una relación está en cuarta forma normal (4FN) si cumple la 3FN y no tiene atributos multievaluados. Una dependencia multievaluada tiene lugar cuando el valor de un atributo determina un conjunto de valores múltiples.

La cuarta forma normal proporciona una base teórica para un principio que es intuitivamente obvio: no se deben combinar en una sola relación grupos repetitivos independientes.

Supongamos por ejemplo que los productos que vende una compañía vienen en múltiples tamaños de paquetes, que son proporcionados por múltiples proveedores y que todos los proveedores proporcionan todos los tamaños. Se podría decir en este caso que el producto multidetermina al tamaño de paquete y Proveedor.

Este tipo de relaciones da lugar a una enorme duplicación de valores de datos, a causa de las dependencias multievaluadas. La aplicación de la 4FN debería eliminar dichas dependencias. La 4FN establece, informalmente, que las dependencias multievaluadas deben dividirse en relaciones separadas. (Se puede encontrar una solución poniendo todos los atributos multievaluados en relaciones formadas por ellos mismos, junto con la clave a la cual se aplican los valores de los atributos.)

Lo importante que hay que comprender sobre la 4FN es que entra en juego solo si hay valores múltiples para los atributos. (Si cada producto en el ejemplo considerado, solo tuviera un tamaño de paquete o un solo proveedor, no se aplicaría esta forma normal).

Conclusiones

La normalización es una técnica que se ha desarrollado para obtener estructuras de datos eficientes, garantizando un buen diseño lógico de la base de datos. Es decir, se utiliza para mejorar el esquema, de modo que éste satisfaga ciertas restricciones que eviten la duplicidad de datos, y garantiza que el esquema resultante esté más próximo al modelo de la empresa, sea consistente, con la mínima redundancia y la máxima estabilidad.

La normalización es una ayuda muy útil en el proceso de diseño de las bases de datos, pero conviene señalar que no es una panacea. Hay que tener en cuenta que las formas normales no son prescripciones para la creación de un modelo de datos "correcto". Un modelo de datos podría llegar a estar perfectamente normalizado, pero podría proporcionar las respuestas tan despacio y de forma tan complicada que el sistema de base de datos construido sobre él resulte inoperativo.

No hay que olvidar que al descomponer una relación penalizamos las consultas, provocando una pérdida de eficiencia en las mismas. Aunque, en general, se aconseja llevar los esquemas relacionales al menos a 3FN, existen ciertos casos en los que, una vez realizada la descomposición, exigencias de eficiencia muy estrictas obligan a llevar a cabo el proceso inverso, es decir, una desnormalización, combinando las relaciones hasta dejarlas en formas normales anteriores.

Por lo tanto, hay que poner en la balanza hasta dónde conviene normalizar para que el resultado sea un modelo de datos eficiente y efectivo, aunque no cabe duda que con las tres primeras formas normales las probabilidades de obtener este resultado son muy altas.

Bibliografía

Este tema se ha elaborado tomando como principales referencias los textos siguientes.

Connolly T. y Begg C. (2005). *Database Systems. A Practical Approach to Design, Implementation and Management*. Addison-Wesley. Segunda edición.

Date C. J. (2001). *Introducción a los sistemas de bases de datos*. Prentice-Hall. Séptima edición.

Riordan R (2000). *Diseño de bases de datos relacionales con Access y SQL Server*, Mc-Graw Hill.