

(2.7) normalización

(2.7.1) problemas del esquema relacional

Una vez obtenido el esquema relacional resultante del esquema entidad/relación que representa la base de datos, normalmente tendremos una buena base de datos. Pero otras veces, debido a fallos en el diseño o a problemas indetectables, tendremos un esquema que puede producir una base de datos que incorpore estos problemas:

- **Redundancia.** Se llama así a los datos que se repiten continua e innecesariamente por las tablas de las bases de datos. Cuando es excesiva es evidente que el diseño hay que revisarlo, es el primer síntoma de problemas y se detecta fácilmente.
- **Ambigüedades.** Datos que no clarifican suficientemente el registro al que representan. Los datos de cada registro podrían referirse a más de un registro o incluso puede ser imposible saber a qué ejemplar exactamente se están refiriendo. Es un problema muy grave y difícil de detectar.
- **Pérdida de restricciones de integridad.** Normalmente debido a **dependencias funcionales**. Más adelante se explica este problema. Se arreglan fácilmente siguiendo una serie de pasos concretos.
- **Anomalías en operaciones de modificación de datos.** El hecho de que al insertar un solo elemento haya que repetir tuplas en una tabla para variar unos pocos datos. O que eliminar un elemento suponga eliminar varias tuplas necesariamente (por ejemplo que eliminar un cliente suponga borrar seis o siete filas de la tabla de clientes, sería un error muy grave y por lo tanto un diseño terrible).

El principio fundamental reside en que las tablas deben referirse a objetos o situaciones muy concretas, relacionados exactamente con elementos reconocibles por el sistema de información de forma inequívoca. Cada fila de una tabla representa inequívocamente un elemento reconocible en el sistema. Lo que ocurre es que conceptualmente es difícil agrupar esos elementos correctamente.

En cualquier caso la mayor parte de problemas se agravan si no se sigue un modelo conceptual y se decide crear directamente el esquema relacional. En ese caso el diseño tiene una garantía casi asegurada de funcionar mal.

Cuando aparecen los problemas enumerados entonces se les puede resolver usando reglas de normalización. Estas reglas suelen forzar la división de una tabla en dos o más tablas para arreglar ese problema.

(2.7.2) formas normales

Las formas normales se corresponde a una teoría de normalización iniciada por el propio **Codd** y continuada por otros autores (entre los que destacan **Boyce** y **Fagin**). Codd definió en 1970 la primera forma normal, desde ese momento aparecieron la segunda, tercera, la Boyce-Codd, la cuarta y la quinta forma normal.

Una tabla puede encontrarse en primera forma normal y no en segunda forma normal, pero no al contrario. Es decir los números altos de formas normales son más restrictivos (la quinta forma normal cumple todas las anteriores).

La teoría de formas normales es una teoría absolutamente matemática, pero en el presente manual se describen de forma más intuitiva.

Hay que tener en cuenta que muchos diseñadores opinan que basta con llegar a la forma Boyce-Codd, ya que la cuarta, y sobre todo la quinta, forma normal es polémica. Hay quien opina que hay bases de datos peores en quinta forma normal que en tercera. En cualquier caso debería ser obligatorio para cualquier diseñador llegar hasta la forma normal de Boyce-Codd.

(2.7.3) primera forma normal (1FN)

Es una forma normal inherente al esquema relacional. Es decir toda tabla realmente relacional la cumple.

Se dice que una tabla se encuentra en primera forma normal si impide que un atributo de una tupla pueda tomar más de un valor. La tabla:

TRABAJADOR		
DNI	Nombre	Departamento
12121212A	Andrés	Mantenimiento
12345345G	Andrea	Dirección Gestión

Visualmente es una tabla, pero no una tabla relacional (lo que en terminología de bases de datos relacionales se llama **relación**). No cumple la primera forma normal.

Sería primera forma normal si los datos fueran:

TRABAJADOR		
DNI	Nombre	Departamento
12121212A	Andrés	Mantenimiento
12345345G	Andrea	Dirección
12345345G	Andrea	Gestión

Esa tabla sí está en primera forma normal.

(2.7.4) dependencias funcionales

dependencia funcional

Se dice que un conjunto de atributos (**Y**) depende funcionalmente de otro conjunto de atributos (**X**) si para cada valor de **X** hay un único valor posible para **Y**. Simbólicamente se denota por **X→Y**.

Por ejemplo el **nombre** de una persona depende funcionalmente del **DNI**; es decir para un DNI concreto sólo hay un nombre posible. En la tabla del ejemplo anterior, el departamento no tiene dependencia funcional, ya que para un mismo DNI puede haber más de un departamento posible. Pero el nombre sí que depende del DNI.

Al conjunto **X** del que depende funcionalmente el conjunto **Y** se le llama **determinante**. Al conjunto **Y** se le llama **implicado**.

dependencia funcional completa

Un conjunto de atributos (**Y**) tiene una dependencia funcional completa sobre otro conjunto de atributos (**X**) si **Y** tiene dependencia funcional de **X** y además no se puede obtener de **X** un conjunto de atributos más pequeño que consiga una dependencia funcional de **Y** (es decir, no hay en **X** un determinante formado por atributos más pequeños).

Por ejemplo en una tabla de clientes, el conjunto de atributos formado por el **nombre** y el **dni** producen una dependencia funcional sobre el atributo **apellidos**. Pero no es plena ya que el **dni** individualmente, también produce una dependencia funcional sobre **apellidos**. El dni sí produce una dependencia funcional completa sobre el campo apellidos.

Una dependencia funcional completa se denota como $X \twoheadrightarrow Y$

dependencia funcional elemental

Se produce cuando **X** e **Y** forman una dependencia funcional completa y además **Y** es un único atributo.

dependencia funcional transitiva

Es más compleja de explicar, pero tiene también utilidad. Se produce cuando tenemos tres conjuntos de atributos **X**, **Y** y **Z**. **Y** depende funcionalmente de **X** ($X \rightarrow Y$), **Z** depende funcionalmente de **Y** ($Y \rightarrow Z$). Además **X** no depende funcionalmente de **Y** ($Y \not\rightarrow X$). Entonces ocurre que **X** produce una dependencia funcional transitiva sobre **Z**.

Esto se denota como: ($X \twoheadrightarrow Z$)

Por ejemplo si **X** es el atributo **Número de Clase** de un instituto, e **Y** es el atributo **Código Tutor**. Entonces $X \rightarrow Y$ (el tutor depende funcionalmente del número de clase). Si **Z** representa el **Código del departamento**, entonces $Y \rightarrow Z$ (el código del departamento depende funcionalmente del código tutor, cada tutor sólo puede estar en un departamento). Como ocurre que $Y \not\rightarrow X$ (el código de la clase no depende funcionalmente del código tutor, un código tutor se puede corresponder con varios códigos de clase). Entonces $X \twoheadrightarrow Z$ (el código del departamento depende transitivamente del código de la clase).

(2.7.5) segunda forma normal (2FN)

Ocurre si una tabla está en primera forma normal y además cada atributo que no sea clave, depende de forma funcional completa respecto de cualquiera de las claves. Toda la clave principal debe hacer dependientes al resto de atributos, si hay atributos que depende sólo de parte de la clave, entonces esa parte de la clave y esos atributos formarán otra tabla. Ejemplo:

ALUMNOS				
DNI	Cod Curso	Nombre	Apellido1	Nota
12121219A	34	Pedro	Valiente	9
12121219A	25	Pedro	Valiente	8
3457775G	34	Ana	Fernández	6
5674378J	25	Sara	Crespo	7
5674378J	34	Sara	Crespo	6

Suponiendo que el DNI y el código de curso formen una clave principal para esta tabla, sólo la nota tiene dependencia funcional completa. El nombre y los apellidos dependen de forma completa del DNI. La tabla no es 2FN, para arreglarlo:

ALUMNOS		
DNI	Nombre	Apellido1
12121219A	Pedro	Valiente
3457775G	Ana	Fernández
5674378J	Sara	Crespo

ASISTENCIA		
DNI	Cod Curso	Nota
12121219A	34	9
12121219A	25	8
3457775G	34	6
5674378J	25	7
5674378J	34	6

(2.7.6) tercera forma normal (3FN)

Ocurre cuando una tabla está en 2FN y además ningún atributo que no sea clave depende transitivamente de las claves de la tabla. Es decir no ocurre cuando algún atributo depende funcionalmente de atributos que no son clave.

Ejemplo:

ALUMNOS				
DNI	Nombre	Apellido1	Cod Provincia	Provincia
12121349A	Salvador	Velasco	34	Palencia
12121219A	Pedro	Valiente	34	Palencia
3457775G	Ana	Fernández	47	Valladolid
5674378J	Sara	Crespo	47	Valladolid
3456858S	Marina	Serrat	08	Barcelona

La Provincia depende funcionalmente del código de provincia, lo que hace que no esté en 3FN. El arreglo sería:

ALUMNOS			
DNI	Nombre	Apellido1	Cod Provincia
12121349A	Salvador	Velasco	34
12121219A	Pedro	Valiente	34
3457775G	Ana	Fernández	47
5674378J	Sara	Crespo	47
3456858S	Marina	Serrat	08

PROVINCIA	
Cod Provincia	Provincia
34	Palencia
47	Valladolid
08	Barcelona

(2.7.7) forma normal de Boyce-Codd (FNBC o BCFN)

Ocurre si una tabla está en tercera forma normal y además todo determinante es una clave candidata. Ejemplo:

ORGANIZACIÓN		
Trabajador	Departamento	Responsable
Alex	Producción	Felipa
Arturo	Producción	Martín
Carlos	Ventas	Julio
Carlos	Producción	Felipa
Gabriela	Producción	Higinio
Luisa	Ventas	Eva
Luisa	Producción	Martín
Manuela	Ventas	Julio
Pedro	Ventas	Eva

La cuestión es que un trabajador o trabajadora puede trabajar en varios departamentos. En dicho departamento hay varios responsables, pero cada trabajador sólo tiene asignado uno. El detalle importante que no se ha tenido en cuenta, es que el o la responsable sólo puede ser responsable en un departamento.

Este detalle último produce una dependencia funcional ya que:

Responsable → Departamento

Por lo tanto hemos encontrado un determinante que no es clave candidata. No está por tanto en FNBC. En este caso la redundancia ocurre por mala selección de clave. La redundancia del departamento es completamente evitable. La solución sería:

PERSONAL	
Trabajador	Responsable
Alex	Felipa
Arturo	Martín
Carlos	Julio
Carlos	Felipa
Gabriela	Higinio
Luisa	Eva
Luisa	Martín
Manuela	Julio
Pedro	Eva

RESPONSABLES	
Responsables	Departamento
Felipa	Producción
Martín	Producción
Julio	Ventas
Higinio	Producción
Eva	Ventas

En las formas de Boyce-Codd hay que tener cuidado al descomponer ya que se podría perder información por una mala descomposición

(2.7.8) cuarta forma normal (4FN). dependencias multivaluadas

dependencia multivaluada

Para el resto de formas normales (las diseñadas por **Fagin**, mucho más complejas), es importante definir este tipo de dependencia, que es distinta de las funcionales. Si las funcionales eran la base de la segunda y tercera forma normal (y de la de Boyce-Codd), éstas son la base de la cuarta forma normal.

Una dependencia multivaluada de X sobre Y (es decir $X \twoheadrightarrow Y$), siendo X e Y atributos de la misma tabla, ocurre cuando Y tiene un conjunto de valores bien definidos sobre cualquier valor de X. Es decir, dado X sabremos los posibles valores que puede tomar Y.

Se refiere a posibles valores (en plural) y se trata de que los valores de ese atributo siempre son los mismos según el valor de un atributo y no del otro.

Ejemplo:

Nº Curso	Profesor	Material
17	Eva	1
17	Eva	2
17	Julia	1
17	Julia	2
25	Eva	1
25	Eva	2
25	Eva	3

La tabla cursos, profesores y materiales del curso. La tabla está en FNBC ya que no hay dependencias transitivas y todos los atributos son clave sin dependencia funcional hacia ellos. Sin embargo hay redundancia. Los materiales se van a repetir para cualquier profesor dando cualquier curso, ya que los profesores van a utilizar todos los materiales del curso (de no ser así no habría ninguna redundancia).

Los materiales del curso dependen de forma multivaluada del curso y no del profesor en una dependencia multivaluada (no hay dependencia funcional ya que los posibles valores son varios). Para el par **Nº de curso** y **Profesor** podemos saber los materiales; pero lo sabemos por el curso y no por el profesor.

cuarta forma normal

Ocurre esta forma normal cuando una tabla está en forma normal de Boyce Codd y toda dependencia multivaluada es una dependencia funcional. Para la tabla anterior la solución serían dos tablas:

Nº Curso	Material
17	1
17	2
25	1
25	2
25	3

Nº Curso	Profesor
17	Eva
17	Julia
25	Eva

Un teorema de Fagin indica cuando hay tres pares de conjuntos de atributos **X**, **Y** y **Z** si ocurre **X->>Y** y **X->>Z** (**Y** y **Z** tienen dependencia multivaluada sobre **X**), entonces las tablas **X**, **Y** y **X**, **Z** reproducen sin perder información lo que poseía la tabla original. Este teorema marca la forma de dividir las tablas hacia una 4FN

(2.7.9) quinta forma normal (5FN)

dependencias de JOIN o de reunión

Una **proyección** de una tabla es la tabla resultante de tomar un subconjunto de los atributos de una tabla (se trata de la operación proyección, **Π** , del álgebra relacional). Es decir una tabla formada por unas cuantas columnas de la tabla original.

La operación **JOIN** procedente también del álgebra relacional, consiste en formar una tabla con la unión de dos tablas. La tabla resultante estará formada por la combinación de todas las columnas y filas de ambas, excepto las columnas y filas repetidas.

Se dice que se tiene una tabla con **dependencia de unión (o de tipo JOIN)** si se puede obtener esa tabla como resultado de combinar mediante la operación JOIN varias proyecciones de la misma.

quinta forma normal o forma normal de proyección-unión

Ocurre cuando una tabla está en 4FN y cada dependencia de unión (JOIN) en ella es implicada por las claves candidatas.

Es la más compleja y polémica de todas. Polémica pues no está claro en muchas ocasiones está muy claro que el paso a 5FN mejora la base de datos. Fue definida también por Fagin.

Es raro encontrarse este tipo de problemas cuando la normalización llega a 4FN. Se deben a restricciones semánticas especiales aplicadas sobre la tabla.

Ejemplo:

Proveedor	Material	Proyecto
1	1	2
1	2	1
2	1	1
1	1	1

Indican códigos de material suministrado por un proveedor y utilizado en un determinado proyecto. Así vista la tabla, no permite ninguna proyección en la que no perdamos datos.

Pero si ocurre una restricción especial como por ejemplo: *Cuando un proveedor nos ha suministrado alguna vez un determinado material, si ese material aparece en otro proyecto, haremos que el proveedor anterior nos suministre también ese material para el proyecto.*

Eso ocurre en los datos como el proveedor número 1 nos suministró el material número 1 para el proyecto 2 y en el proyecto 1 utilizamos el material 1, aparecerá la

tupla proveedor 1, material 1 y proyecto 1. Si un nuevo proyecto necesitara el material 1, entonces habrá que pedirlo a los proveedores 1 y 2 (ya que en otros proyectos les hemos utilizado)

La dependencia de reunión que produce esta restricción es muy difícil de ver ya que es lejana. Para esa restricción esta proyección de tablas sería válida:

<u>Proveedor</u>	<u>Material</u>
1	1
1	2
2	1

<u>Material</u>	<u>Proyecto</u>
1	2
2	1
1	1

<u>Proveedor</u>	<u>Proyecto</u>
1	2
1	1
2	1

Esa descomposición no pierde valores en este caso, sabiendo que si el proveedor nos suministra un material podremos relacionarle con todos los proyectos que utilizan ese material.

Resumiendo, una tabla no está en quinta forma normal si hay una descomposición de esa tabla que muestre la misma información que la original y esa descomposición no tenga como clave la clave original de la tabla.