

Dependencias Funcionales

Dependencia funcional

Lo primero consiste en revisar algunas definiciones:

Def. 1: Una dependencia funcional es una relación muchos a uno desde un conjunto de atributos a otro que tiene una relación.

Def. 2: Sea R una relación y sean X e Y subconjuntos arbitrarios del conjunto de atributos de R . Cuando se dice que Y es funcionalmente dependiente de X -en símbolos $X \rightarrow Y$ - sí y sólo sí cada valor de X en R está asociado con, precisamente, un valor de Y en R .

En otras palabras, cuando quiera que dos tuplas de R coinciden sobre un mismo valor de X , ellas también coinciden sobre un mismo valor de Y .

Las definiciones anteriores explicitan que para una relación R es posible considerar un amplio conjunto de dependencias funcionales entre los atributos que la componen. Esto es obvio si consideramos para la segunda definición que X es una clave candidata -más aún si es una clave principal- e Y son todos los atributos de la relación.

Estas dependencias se establecen para todos los posibles valores legales de los atributos considerados. Esto significa que puede darse el caso que todas las tuplas que actualmente existen en la relación cumplan con una determinada restricción funcional, pero eso es algo que en la realidad no ocurre y es en este sentido que se habla de los valores legales de los atributos, *todos los posibles en la realidad*.

Dependencias Triviales y No triviales

Una forma obvia de reducir el tamaño del conjunto de dependencias es eliminar las dependencias triviales. Una dependencia es *trivial* si es imposible que no pueda ser satisfecha. Esto ocurre, en efecto, si y sólo si el conjunto a la derecha es un subconjunto del que está a la izquierda. Las dependencias no triviales son todo aquel conjunto que es distinto de subconjunto de las triviales.

Dependencia funcional completa

Se dice que el atributo Y de la relación R es por completo dependiente funcionalmente del atributo X de la relación R si depende funcionalmente de X y no depende funcionalmente de ningún subconjunto propio de X (es decir, no existe un subconjunto propio Z de los atributos componentes de X tales que Y sea funcionalmente dependiente de Z).

Propiedades

Sean A, B, C y D son subconjuntos arbitrarios del conjunto de atributos que tiene la relación R , y escribir AB significa la Unión de A y B . Entonces tenemos:

Reglas de inferencia de Armstrong

1. Reflexividad: Si B es un subconjunto de A , entonces $A \rightarrow B$.
2. Aumentación: Si $A \rightarrow B$, entonces $AC \rightarrow BC$.
3. Transitividad: Si $A \rightarrow B$ y $B \rightarrow C$, entonces $A \rightarrow C$.
4. Autodeterminación: $A \rightarrow A$.
5. Descomposición: Si $A \rightarrow BC$, entonces $A \rightarrow B$ y $A \rightarrow C$.
6. Unión: Si $A \rightarrow B$ y $A \rightarrow C$, entonces $A \rightarrow BC$.
7. Composición: Si $A \rightarrow B$ y $C \rightarrow D$, entonces $AC \rightarrow BD$.

Teorema de Unificación General: \cup :unión y $-$:diferencia.

Si $A \rightarrow B$ y $C \rightarrow D$, entonces $A \cup (C - B) \rightarrow BD$

Dependencia Multivaluada

Def.-3: Dada una relación R con los atributos A, B y C, la dependencia multivaluada (DMV) $R.A \twoheadrightarrow R.B$ se cumple en R si y sólo si el conjunto de valores de B correspondiente a un par dado (valor de A, valor de C) en R depende sólo del valor de A y es independiente del valor de C

Dependencias funcionales de las claves

Una *clave candidata* consiste de un conjunto de atributos K (no vacío) de la relación R que satisface las siguientes propiedades, independientes del tiempo:

1. *Unicidad:* en cualquier momento dado, no existen dos tuplas en R con el mismo valor de K.
2. *Minimalidad:* si K es compuesto, no será posible eliminar ningún componente de K sin destruir la propiedad de unicidad.

Nótese que la relación tiene por lo menos una clave candidata, porque las relaciones no contienen tuplas repetidas. Ahora bien, del conjunto de las claves candidatas de una relación dada, se elige una y sólo una como *clave primaria* de esa relación; las demás, si existen se llaman *claves alternativas*.

Revisando las definiciones anteriores y comparándolas con la definición de dependencia funcional podemos decir que siempre existirá una dependencia entre cualquier clave candidata y los atributos que no son o no componen dicha clave, pero que también pertenecen a la relación.

Ejemplos de Dependencias Funcionales

Las dependencias funcionales son representaciones semánticas de las relaciones que existen entre los datos en una realidad. Un buen ejemplo de esto es el *nombre* de una **persona**; el cual siempre dependerá del *rut* de esa persona; ya que aunque existiesen dos personas con el mismo nombre, ellas siempre tendrán distinto rut.

Pero, como se señaló, las dependencias funcionales reflejan enlaces semánticos permanentes entre datos de un diseño. Y es en este último sentido que podríamos pensar que el ejemplo entregado anteriormente puede no ser un ejemplo de una dependencia funcional dentro de un diseño, ya que la existencia o no de alguna de ellas es una decisión del diseñador.

En relación a lo anterior, podemos pensar en la situación de un **alumno** de ésta universidad: podemos considerar que su *rut* y su *nombre* dependen funcionalmente de su *número de matrícula* y la dependencia del *nombre* al *rut* obviarla.

Con la situación anterior podemos también ejemplificar, forzando la situación, lo que es una dependencia funcional **completa**: por ejemplo, supongamos para un **alumno** que el *nombre* lo hacemos depender del conjunto de atributos compuesto por *rut* y *número de matrícula*; pero, realmente el *nombre* de una persona (en este caso alumno) depende sólo del *rut*, un subconjunto de los atributos que componen nuestro conjunto inicial; el ejemplo anterior derivaría en una dependencia completa, si sólo se hiciera depender el atributo *nombre* del *rut*.

Gráficamente es posible representar una dependencia funcional mediante flechas, esto ya se vio en la definición en que simbólicamente se representa una dependencia de Y con respecto de X mediante la expresión: $X \rightarrow Y$. Por ejemplo para las relaciones de alumno y ramos se tendría los siguientes atributos:

Alumno(NoMatrícula, Nombre, Sexo, FechaNac.)

Ramos(NoMatrícula, CodRamo, NotaFinal)

Con las siguientes dependencias funcionales:

Alumno.NoMatrícula \rightarrow Nombre

Alumno.NoMatrícula \rightarrow Sexo

Alumno.NoMatrícula → FechaNac

(NoMatrícula, CodRamo) → NotaFinal

Dependencias Funcionales y Formas Normales

Procedimiento de normalización adicional

La teoría de la normalización busca que las relaciones cumplan con un cierto conjunto de restricciones explícitas, de modo que sea posible asegurar ciertas propiedades deseables en los datos.

Las formas normales corresponden a un conjunto discreto de restricciones progresivas que hacen que una relación tenga un grado de normalización en el momento que cumple con algún subconjunto de aquellas. El sentido de *progresión* que determinan las formas normales es justificado por las características de las restricciones que ellas imponen, donde, para que una relación acceda a la posibilidad de cumplir el siguiente conjunto de restricciones debe, previamente satisfacer las anteriores. Esto se puede ver desde el punto que hay ciertas relaciones a las cuales les basta con cumplir un conjunto de restricciones y con ello alcanzar un grado de normalización suficiente. Mientras, por otro lado, existen otras relaciones que necesitan cumplir con otro conjunto mayor de restricciones para alcanzar éste estado deseable en sus datos.

Así las formas normales están contenidas unas en otras formando un conjunto de reglas que progresivamente va asegurando calidad en las relaciones. Esto se alcanza mediante un *procedimiento de normalización* en el cual una relación en unacierta forma normal, se puede convertir en una o más relaciones que están en una forma más deseable. Así, el procedimiento lo podemos caracterizar como la reducción sucesiva de un conjunto dado de relaciones a una forma más deseable.

Cabe señalar que el procedimiento es reversible; es decir, siempre es posible tomar la salida del procedimiento (relaciones con un grado de normalización) y convertirlas otra vez en la entrada (antes de cumplir con las restricciones). Esto último es importante ya que significa que no se pierde información durante el proceso de normalización.

Formas Normales

Las dependencias funcionales permiten definir de manera más formal los elementos de normalización del modelo relacional.

Para ello, y entendiendo que no es simple la comprensión del problema de las formas normales y su relación con las dependencias funcionales, se establece el siguiente método de desarrollo explicativo: primero, se entregará la definición que propone C.J.Date en el libro "Introducción a los Sistemas de Bases de Datos", la cual, probablemente necesitará de algún grado de explicación; así, en segundo lugar, se entregará una breve explicación de esa definición.

Primera forma normal

- ☒ Una relación está en *primera forma normal* (1NF) si y sólo si todos los dominios simples subyacentes contienen sólo valores atómicos.

Las dependencias no participan directamente en la normalización para alcanzar la primera forma normal puesto que ésta sólo busca eliminar los grupos repetitivos. Así, el siguiente es un ejemplo de este nivel de normalización sobre el atributo b:

R:		
a	b	c
a1	b1,b2,b3	c1

a2	b4,b5	c2
----	-------	----

R1:		
a	b	c
a1	b1	c1
a1	b2	c1
a1	b3	c1
a2	b4	c2
a2	b5	c2

se puede apreciar que b presenta repeticiones de ocurrencias para R, al normalizar obtenemos R1, en la cual se han eliminado estas repeticiones.

Segunda forma normal

- ☒ Una relación está en *segunda forma normal* (2NF) si y sólo si está en 1NF y todos los atributos no clave dependen por completo de la clave primaria.

Esta forma aboga por relaciones en que solamente existan dependencias completas, eliminando, mediante proyección, las que estando en 1º forma normal no cumplan esa condición. Por ejemplo:

R(a,b,c,d,e) con DF: {a,b} \rightarrow c, a \rightarrow d, a \rightarrow e

donde las dos últimas no son completas con respecto de la clave primaria, así es necesaria una proyección que dé como resultado dos o más relaciones que tengan esta característica, por ejemplo:

R1(a,b,c) con DF: {a,b} \rightarrow c, y

R2(a,d,e) con DF: a \rightarrow d, a \rightarrow e

Tercera forma normal

- ☒ Una relación está en *tercera forma normal* (3NF) si y sólo si esta en 2NF y todos los atributos no clave dependen de manera no transitiva de la clave primaria.

Busca, para relaciones en 2º forma normal, aquellas en que existan dependencias transitivas con respecto de la clave primaria. Así, cualquier relación que tenga este tipo de dependencia, debe ser proyectada de modo que sea eliminada. Por ejemplo:

R (a,b,c) con DF: a \rightarrow b, a \rightarrow c, b \rightarrow c

Se puede apreciar que la dependencia funcional a \rightarrow c es transitiva, ya que puede ser expresada mediante las otras dos que existen, esto es: a \rightarrow b y b \rightarrow c. Para eliminar esto proyectamos sobre a,b y sobre b,c

R1(a,b) con DF: a \rightarrow b

R2(b,c) con DF: b \rightarrow c

Forma normal de Boyce/Codd

- ☑ Una relación está en *forma normal Boyce/Codd* (BCNF) si y sólo si todo determinante es una clave candidata.

Se debe definir el concepto de *determinante* como un atributo del cual depende funcionalmente (por completo) algún otro atributo. Por ejemplo, en el caso anterior, *a* es un determinante para todos los atributos de *R1*, también *b* es el determinante de todos los atributos de *R2*.

Para la forma normal Boyce/Codd los únicos determinantes son las claves candidatas, es decir los atributos sólo puede depender de las claves candidatas. Veamos un caso en que eso no ocurra, y sea necesaria una transformación para normalizarla según la forma Boyce/Codd:

$R(a,b,c)$ donde existen las siguientes DF $\{a,b\} \rightarrow c$ y $c \rightarrow b$. Esto se normaliza en Boyce/Codd como las siguientes dos relaciones:

$R1(a,c)$ con DF $a \rightarrow c$, y

$R2(c,b)$ con DF $c \rightarrow b$

Cuarta Forma Normal

- ☑ Una relación *R* está en *cuarta forma normal* (4NF) si y sólo si, siempre que existe una Dependencia Multivaluada en *R*, digamos $A \twoheadrightarrow B$, todos los atributos de *R* dependen también funcionalmente de *A*. En otras palabras, las únicas dependencias (funcionales o multivaluadas) en *R* son de la forma $K \rightarrow X$ (o sea, una dependencia funcional con respecto a una clave candidata *K* de algún otro atributo *X*). O lo que es equivalente: *R* está en 4NF si está en BCNF y todas las dependencias multivaluadas en *R* son de hecho dependencias funcionales.

Por ejemplo, si tenemos la siguiente relación, con dependencias multivaluadas:

$R(a,b,c)$ donde existen las DMV $a \twoheadrightarrow b$ y $a \twoheadrightarrow c$, entonces la normalización en cuarta forma normal nos lleva a transformar esta relación en otras en que todas las dependencias multivaluadas sean dependencias funcionales, esto sólo lo logramos al proyectarlas sobre distintas relaciones, por ejemplo:

$R1(a,b)$ con DF $a \rightarrow b$, y

$R2(a,c)$ con DF $a \rightarrow c$

Quinta forma normal

- ☑ Una relación está en *quinta forma normal* (5NF) -llamada también forma normal de proyección/reunión (PJ/NF)- si y sólo si toda dependencia de reunión en *R* es una consecuencia de las claves candidatas de *R*.

Introducción a la Automatización del Proceso de normalización

Una de las ventajas de cualquier formalización teórica es la posibilidad de crear herramientas de apoyo a procesos que, basados en esta formalización, generen resultados consistentes.

Aplicando lo anterior podemos vislumbrar una conceptualización tal que para el terreno de las bases de datos relacionales sea factible la automatización del procedimiento de normalización adicional.

Esta conceptualización se basa en que si tenemos un esquema de base de datos *R* lo podemos llevar, mediante un proceso de normalización, a un esquema *S*, en lo que sería una *adecuada representación* de *R* en términos de las restricciones impuestas por *S*.

Esta transformación de *R* en *S* se logra, principalmente, mediante el uso de dos tipos de descomposiciones. Horizontales que usa el operador de *selección* y verticales mediante el uso de operador de *proyección*.

También, cualquier instancia *R* debe ser posible reconstruirla desde *S* mediante la aplicación de una transformación "inversa" a la que permitió llevar *R* a *S*.

Dada la formalización algebraica de la teoría de bases de datos relacionales, es posible expresar

las relaciones que existen entre un esquema R y su descomposición adecuada equivalente S , en términos de requerimientos formales y operaciones no ambiguas sobre R , de manera que éste cumpla con las restricciones necesarias impuestas por S . Un ejemplo de transformaciones de ese tipo, corresponde a las anteriormente descritas.

Técnicas de normalización

Existen dos técnicas principales para la normalización de esquema mediante descomposición: *análisis* y *síntesis*. Ambos reciben como entrada el diseño de una relación, y el conjunto de las dependencias funcionales definidas sobre ella. Pero las dos técnicas difieren en la estrategia que usan para obtener el esquema normalizado.

El algoritmo de análisis genera el esquema normalizado mediante la eliminación de las causas que impiden la normalización. Si una relación no satisface la deseada forma normal por causa de ciertas dependencias, ella es descompuesta en dos o más relaciones en base a tal dependencia. El proceso es iterado hasta que todas las relaciones están normalizadas. Así, a cada paso el anterior esquema es cambiado en uno nuevo con más relaciones.

Por ejemplo, si una relación como R con sus dependencias funcionales y queremos llevarla a tercera forma normal, los pasos que se ejecutan serán los siguientes:

$R(a,b,c,d,e,f)$ con las DF $a \rightarrow b$, $b \rightarrow c$, $c \rightarrow d$, $a \rightarrow e$ y $a \rightarrow f$

Paso 1, primera iteración:

$R12(a,b,c,e,f)$ con las DF $a \rightarrow b$, $b \rightarrow c$, $a \rightarrow e$ y $a \rightarrow f$

$R3(c,d)$ con las DF $c \rightarrow d$

Se puede ver que mientras $R3$ está en tercera forma normal $R12$ aún mantiene una dependencia transitiva, iteramos nuevamente para $R12$.

Paso 2, segunda iteración:

$R1(a,b,e,f)$ con las DF $a \rightarrow b$, $a \rightarrow e$ y $a \rightarrow f$

$R2(b,c)$ con las DF $b \rightarrow c$

Así, las tres relaciones resultantes $R1, R2, R3$ están en tercera forma normal.

Por otro lado, la síntesis actúa considerando las dependencias que existen, reduciéndolas a una cobertura estándar, y particionándolas en base a un criterio específico. Una relación, entonces, está asociada con cada elemento de la partición, conteniendo todos los atributos que están involucrados en dependencias con ese elemento. La partición es generada como una forma en que la relación correspondiente puede agrupar funcionalmente los atributos. Así, al final la relación es sintetizada sin generar relaciones intermedias.

Por ejemplo, consideremos la misma relación R que utilizamos en el caso del análisis. Donde lo primero que haremos será una partición en el conjunto de las dependencias funcionales y en base a eso e inspección se sintetizarán las nuevas relaciones:

La partición nos entrega las siguientes dependencias funcionales, y con ellas es posible formar tres relaciones normalizadas:

$a \rightarrow b$

$a \rightarrow e$ que forman $R1(a,b,e,f)$

$a \rightarrow f$

$b \rightarrow c$ que conforma $R2(b,c)$

$c \rightarrow d$ que, finalmente, permite $R3(c,d)$

Estos dos algoritmos, someramente presentados, entregan una visión de las posibilidades de algoritmización del problema de la normalización gracias a la especificación de dependencias funcionales.