CC42A/CC55A - BASES DE DATOS
Profesor: Claudio Gutiérrez
Auxiliar: Mauricio Monsalve

Dependencias funcionales

1 El concepto de dependencia funcional

1.1 El concepto de dependencia funcional

Hay veces en que los atributos están relacionados entre sí de manera más específica que la de pertenecer a una misma relación. Hay veces en que es posible determinar que un atributo depende de otro *funcionalmente*, como si existiera una función f en el "mundo", tal que $t[A] = f(t[B])$. La función se anotaría como $f : A \to B$, pero como f es desconocida (o sino B sería un atributo derivado), sólo nos quedamos con $A \to B$, la dependencia funcional, que se lee "A determina B".

Formalmente, $X \to Y$ en R se cumple si y sólo si $\forall s, t \in R, s[X] = t[X] \Rightarrow s[Y] = t[Y]$. Esto es análogo a las funciones: $\forall x_1, x_2 \in X, x_1 = x_2 \Rightarrow f(x_1) = f(x_2)$, con $f : X \to Y$.

1.2 Utilidad en el diseño de bases de datos

Las dependencias funcionales son restricciones de integridad sobre los datos. Conocer las dependencias funcionales en el momento del diseño de la base de datos permite crear mecanismos para evitar la redundancia (y los potenciales problemas de integridad que eso conlleva) y mejorar la eficiencia.

1.3 Un ejemplo real

Por ejemplo, sea la siguiente relación: Vehiculo(serie, nombre, motor, carrocería, peso, eficiencia). Aquí, serie es la llave. Por ende, hay sólo un [modelo de] motor por serie, sólo una [forma de] carrocería por serie, sólo un peso por serie y sólo una eficiencia [energética] por serie: nombre = nombre(serie), motor = motor(serie), etcétera. O sea, $serie \to nombre, motor, carrocera, peso, eficiencia$ (la relación es función de su llave; sólo hay una tupla por llave).

Otra observación, que requiere mucho más conocimiento del problema, nos indica que la eficiencia energética del vehículo es una función del motor, la carrocería y el peso. Considerando esto, tenemos que $motor, carrocera, peso \to eficiencia$. ¿Por qué? La eficiencia energética consiste en la distancia que puede recorrer un vehículo por litro, a una velocidad moderadamente alta. La potencia del vehículo reside en el motor: el modelo de motor indica la fuerza que imprime el vehículo. Sin embargo, esta fuerza es contrarrestada por el roce aerodinámico del vehículo, que es una función del roce viscoso del aire (es un dato fijo) y de la forma de la carrocería. Y el peso entrega la masa del vehículo ($masa = 9,8m/s^2 \times peso$). Si se divide la fuerza resultante del vehículo por la masa, se obtiene la aceleración (y en un equilibrio de velocidades se obtiene la eficiencia). Luego, existe una función tal que $eficiencia = eficiencia(motor, carrocera, peso)$. [1]

[1] Siendo más precisos, notemos que $eficiencia = g(motor, carrocera)/peso$. Entonces, este atributo es medianamente derivado. Luego, es posible definir un atributo $efic\_motor\_carro = g(motor, carrocera)$ a partir del cual "eficiencia" se vuelve un atributo derivado. Esta solución evita aún más redundancia, aunque es más idealizada.

1.4 Un ejemplo más sencillo

A veces es fácil encontrar dependencias en un esquema. Esto es un indicado de un mal modelo entidad-relación o de una mala conversión a relacional.

Por ejemplo, sea Película(título, año, estudio, presidente, fono_presidente). Digamos que "título" es llave de la relación (determina todo). Sin embargo, notemos que el presidente de un estudio se puede determinar conociendo el estudio y el año (idealmente). Luego, $estudio, año \rightarrow presidente$. Además, es claro que $presidente \rightarrow fono\_presidente$.

La relación "Película" fue mal modelada desde un principio. En un modelo entidad-relación, "Película", "Estudio" y "Presidente" habrían sido entidades distintas, luego relaciones distintas en el modelo relacional.

1.5 Un ejemplo visual

¿Qué dependencias funcionales se cumplen en esta relación?

A	B	C	D	E	F	G	H
1	a	eth	cdr	cdr	0	0x00	10
2	a	eth	car	cdr	0	0xf0	10
3	b	usb	cdr	car	0	0xff	10
4	b	com	car	car	1	0x68	10
5	c	lpt	cddr	car	1	0xa0	12

Algunas dependencias funcionales fáciles de ver:

- A determina toda la relación (A es llave).
- B determina H.
- C determina B.
- C determina F.
- D,E determina toda la relación (D,E es llave).
- G es llave.
- etcétera (hay varias más).

1.6 ¿Cómo obtener las dependencias funcionales?

La mejor manera de obtenerlas es a través del conocimiento del problema. Es lo más disponible en la fase de diseño de una base de datos. Sin embargo, esto puede tornarse bastante difícil, como en el caso del vehículo (honestamente, esto puede ocurrir cuando la base de datos modela conocimiento técnico, que escapa al sentido común).

Otra manera, relacionada con el ejemplo anterior, es comprobar dependencias funcionales sobre una gran población de datos usando la definición.

2 Demostración de los axiomas de Armstrong

2.1 Ejercicio

Los axiomas de Armstrong son los siguientes:

1. (dependencia trivial) $A \to B \subseteq A$.
2. (aumentación) $A \to B \Rightarrow (A \cup C) \to (B \cup C)$. O más comodamente, $A \to B \Rightarrow AC \to BC$.
3. (transitividad) $A \to B \wedge B \to C \Rightarrow A \to C$.

Demuestre los axiomas de Armstrong.

2.2 Solución

Esto se hace mediante la definición de dependencia funcional.

1) (dependencia trivial)

Sea $A = A_1, ..., A_n$, y sea $B \subseteq A$ tal que $B = B_1, ..., B_m$. Luego, $\forall i < m, \exists j < n, B_i = A_j$. Entonces, sean $s, t \in R$, tal que $A \subseteq esq(R)$ (A es parte del esquema de R). Luego, si $s[A] = t[A] \Leftrightarrow \forall j < n, s[A_j] = t[A_j]$ (por definición[2]). Esto implica, en particular, que $s[B_i] = t[B_i], \forall i < m$.

Entonces se concluye que $A \to B \subseteq A$. $\square$

2) (aumentación)

Sea $A \to B$, $A, B \subseteq esq(R)$. Luego, $\forall s, t \in R, s[A] = t[A] \Rightarrow s[B] = t[B]$. Si exigimos que $s[C] = t[C]$, $\forall s, t \in R, s[A] = t[A] \wedge s[C] = t[C] \Rightarrow s[B] = t[B]$ se sigue cumpliendo (un mero asunto de lógica proposicional). Y sin afectar los valores de verdad, se cumple que $\forall s, t \in R, s[A] = t[A] \wedge s[C] = t[C] \Rightarrow s[B] = t[B] \wedge s[C] = t[C]$ (nuevamente, un mero asunto de lógica). Pero la última implicancia es lo mismo que $AC \to BC$, con lo que se prueba el axioma. $\square$

3) (transitividad)

Sean $A, B, C \subseteq esq(R)$ y las dependencias $A \to B$ y $B \to C$. Por definición, se cumplen:
$A \to B : \forall s, t \in R, s[A] = t[A] \Rightarrow s[B] = t[B]$
$B \to C : \forall s, t \in R, s[B] = t[B] \Rightarrow s[C] = t[C]$
Luego, usando la transitividad de las implicancias ($\Rightarrow$), se obtiene:
$\forall s, t \in R, s[A] = t[A] \Rightarrow s[C] = t[C]$, o sea, $A \to C$. $\square$

3 Una manera más directa de proceder

3.1 Una pequeña propiedad

Sea $A \to BC$ y $B \to D$. Entonces, $A \to ABCD$.

Prueba:
$A \to BC \Rightarrow A \cup A \to BC \cup A \Rightarrow A \to ABC$ (aumentación).
$B \to D \Rightarrow B \cup B \to D \cup B \Rightarrow B \to BD$ (aumentación).
$B \to BD \Rightarrow B \cup AC \to BD \cup AC \Rightarrow ABC \to ABCD$ (aumentación).
$A \to ABC \wedge ABC \to ABCD \Rightarrow A \to ABCD$ (transitividad). $\square$

[2] Esto es igual a la comparación de vectores: dos vectores son iguales si y sólo si cada componente posee el mismo valor: $\vec{X} = \vec{Y} \Leftrightarrow \forall j, X_j = Y_j$.

3.2 Ejemplo de uso de la regla anterior

Sea $R(A, B, C)$ y $F = \{A \to B, C \to AB, B \to BC\}$. Calcular F+.

(A modo de nota, F+ es la clausura de F. Es el conjunto de todas las dependencias funcionales que se pueden deducir de F.)

Solución:
$A+ : A \to AB(A \to B)$
$A \to AB \to ABC(B \to BC) - llave -$
$B+ : B \to BC(B \to BC - idem-)$
$B \to BC \to ABC(C \to AB) - llave -$
$C+ : C \to ABC(C \to AB) - llave -$
$AB+ : AB \to ABC$
$AC+ : AC \to ABC$
$BC+ : BC \to ABC$
$ABC+ : ABC \to ABC$

Como se puede notar, resolver dependencias funcionales aumentando el lado derecho (el determinado) de las dependencias, asegura el avance de la resolución. En efecto, cada dependencia se puede usar una sola vez por la clausura de cada atributo. Además, la resolución del problema se simplifica bastante.

4 Ejercicios

4.1 Problema

Sea $R(A, B, C, D, E)$ y $DF = \{A \to BC, CD \to E, B \to D, E \to A\}$. Determinar las llaves candidato (minimales) de R.

4.2 Solución

Calcularemos las clausuras de todos los atributos (descartando cualquier superllave). Para dar legibilidad a la respuesta, enumero las dependencias funcionales de DF:

$$DF = \{A \to BC(1), CD \to E(2), B \to D(3), E \to A(4)\}$$

Así, cada vez que use una dependencia funcional, la mencionaré con un número.

A+: $A \to ABC(1) \to ABCD(3) \to ABCDE(2)$. Luego A es llave candidato.
B+: $B \to BD(3)$.
C+: $C \to C$.
D+: $D \to D$.
E+: $E \to A(4) \to ABCDE(A+)$. E es llave candidato.
Para no considerar superllaves, no usaré A ni E. Sólo combinaré B, C y D.
BC+: $BC \to BCD(3) \to BCDE(2) \to ABCDE(4)$. BC es llave candidato.
BD+: $BD \to BD$.
CD+: $CD \to CDE(2) \to ABCDE(E+)$. CD es llave candidato.
No puedo considerar más combinaciones.
Luego, A, E, BC y CD son llaves candidato.

4.3 Problema

Sea $R(A, B, C, D, E, F)$ y $DF = \{BD \to E, CD \to A, E \to C, B \to D\}$. ¿Cuáles son las llaves minimales?

4.4 Solución

Aquí conviene hacer una observación: todo atributo que no es determinado por otro es parte de la llave minimal (¿por qué?). Observando el lado derecho de cada dependencia, vemos que ni B ni F son determinados por otros atributos (F ni siquiera es parte de alguna dependendencia funcional). Luego, empezar por BF+ puede ser una buena estrategia.

Otra vez voy a enumerar las dependencias, para facilitar la lectura:

$$DF = \{BD \to E(1), CD \to A(2), E \to C(3), B \to D(4)\}$$

BF+: $BF \to BDF(4) \to BDEF(1) \to BCDEF(3) \to ABCDEF(4)$. Luego, BF es llave minimal.

Como B y F son atributos que deben ser parte de toda llave minimal, BF es la única llave minimal de R.

5 Formas normales

La forma normal de una relación indica la calidad de su diseño en cuanto a la redundancia de información evitada. Y su definición está basada en las dependencias, sean funcionales, multivaluadas, de join, etc. En el curso se trabajará sólo con dependencias funcionales (debido a una relación entre eficiencia y control de redundancia).

5.1 Primera forma normal: 1FN

Una relación está en primera forma normal cuando todos sus atributos son atómicos. Como se indicó en clases, siempre estamos en 1FN (por lo menos).

5.2 Segunda forma normal: 2FN

Todos los atributos no primos dependen de la totalidad de la llave (y no de un subconjunto de esta). Nota: por ”la llave” se entiende de ”cualquier llave minimal” o bien ”todas las llaves minimales”.

5.3 Tercera forma normal: 3FN (importante)

Un modelo de datos relacional en tercera forma normal se considera de buena calidad. 3FN garantiza un buen equilibrio entre eficiencia y control de redundancia.

Un esquema R está en 3FN si $\forall X \to Y$ no trivial:

- X es superllave (contiene a alguna llave candidato)
- Y es atributo primo (es parte de alguna llave candidato)

Es importante destacar es necesario conocer todas las llaves candidato o minimales.

5.4 Forma normal de Boyce-Codd: FNBC (importante)

Una forma normal ideal, el máximo control de redundancia mediante dependencias funcionales. La idea consiste en que cada atributo depende de sólo de la llave (en su totalidad, no de una parte).

Un esquema R está en FNBC si $\forall X \rightarrow Y$ no trivial, X es superllave (contiene a alguna llave candidato).

5.5 Cuarta forma normal: 4FN

Es como FNBC, pero con dependencias multivaluadas.

Un esquema R está en 4FN si $X \twoheadrightarrow Y$ no trivial, X es superllave. Pero nótese que, como X es superllave, $X \rightarrow Y$. Luego, 4FN exige que toda dependencia multivaluada sea en realidad una dependencia funcional.

5.6 Quinta forma normal: 5FN

Se trata de producto-reunión (join). Una relación tal que $R = A * B * C$ [3], todas relaciones con distintos esquemas, NO está en 5FN. Una relación está en 5FN cuando no se puede dividir y volver a reconstruir sin pérdida de información.

5.7 Lo que interesa

Interesa mucho el dominio de la tercera forma normal y la forma normal de Boyce-Codd. En particular, la 3FN es la forma normal más alta que asegura preservación de la información (preservación de join) y de las dependencias. Pero si se puede llegar a FNBC, mejor todavía.

Ejercicio propuesto: Demuestre que: $4FN \Rightarrow FNBC \Rightarrow 3FN \Rightarrow 2FN$ (alguna versión de esta pregunta ha aparecido en controles).

6 Normalización

El proceso de normalización es un proceso de descomposición de los esquemas de relación hasta que todas las relaciones alcancen la forma normal deseada. En general, interesa:

1. Determinar las llaves candidato (minimales) de cada relación.
2. Determinar la forma normal de la relación.
3. ¿Está en FNBC? Sino, normalizar.
4. ¿La normalización no asegura la preservación de información y dependencias? Conformarse con 3FN.

Ahora, un par de algoritmos para normalizar. Estos algoritmos aseguran la preservación de la información pero no necesariamente la preservación de las dependencias (ello no siempre se puede lograr con FNBC).

[3] El join natural se anota como * y como $\bowtie$.

6.1 Algoritmo para normalizar en FNBC

De manera informal, el algoritmo es el siguiente:

1. (Calcular F+)
2. Sea un esquema R_i que no está en FNBC: determinar $X \rightarrow Y$ que hace que R_i no esté en FNBC y agregar a la descomposición $(R_i - Y) \cup (XY)$ y remover R_i.
3. Terminar cuando todos los esquemas estén en FNBC.

En resumen, por cada dependencia funcional conflictiva con FNBC, sacar el lado derecho de la relación con problemas y agregar una nueva relación que guarde la dependencia. Esta estrategia de normalización no asegura preservar dependencias, pero sí asegura la recuperación de la información por join.

6.2 Ejemplo

Sea $R(A, B, C, D, E)$ y $F = \{A \rightarrow BC(1), C \rightarrow D(2), B \rightarrow E(3)\}$.

1. R no está en FNBC. Basta ver la dependencia (2); claramente C no es llave de R. Entonces partimos R en R1(A,B,C,E) y R2(C,D). No se ha roto ninguna dependencia.
2. R1 no está en FNBC. Basta ver (3); claramente B no es llave de R. Entonces partimos R1 en R3(A,B,C) y R4(B,E). No se ha roto ninguna dependencia.
3. R2, R3 y R4 están en FNBC, por lo que el algoritmo concluye.

6.3 FNBC infactible (ejercicio)

Uno de los objetivos de diseño es preservar la información (recuperación mediante join) y las dependencias.

Sea $R(A, B, C)$ y $F = \{AB\ toC, C \rightarrow B\}$. ¿En qué forma normal se encuentra? Trate de normalizar a FNBC. ¿Es posible lograr una descomposición de R que esté en FNBC y preserve las dependencias funcionales?

6.4 Algoritmo para normalizar en 3FN

1. Calcular F mínimo (conjunto covertor mínimo, conjunto canónico, equivalente minimal, recubrimiento minimal, etc.).
2. Convertir cada dependencia en una relación $(X \rightarrow Y \Rightarrow R_k(XY))$.
3. Si no está la llave en una relación, agregarla.

La dificultad estriba en obtener el conjunto convertor mínimo de dependencias funcionales. De todas maneras, un conjunto es mínimo cuando:

1. $X \rightarrow Y$, Y es atributo atómico.
2. Remover $X \rightarrow Y$ altera F+.
3. Si $X \rightarrow Y$, $\nexists W \subset X : W \rightarrow Y$.

6.5 Ejemplo

Sea $R(A, B, C, D, E)$ y $F = \{A \to B(1), A \to C(2), C \to D(3), B \to E(4)\}$. F es mínimo.

1. Al notar A+ vemos que: $A \to AB(1) \to ABC(2) \to ABCD(3) \to ABCDE(4)$. Y como A no es determinada por ninguna dependencia funcional, A es la única llave.
2. Tomando las cuatro dependencias, tenemos: $R_1(A, B), R_2(A, C), R_3(C, D), R_4(B, E)$. Como R_1 y R_2 preservan la llave, la normalización ha terminado.

OJO: Sean S y R dos relaciones en 3FN con la misma llave (y todas las dependencias están preservadas). ¿Qué ocurre con $S \cup R$? (Hable de llaves candidato, forma normal, etc.)

7 Y el informe

Lo primero es convertir el diagrama E/R en relacional. Este es un paso bastante trivial pero puede tomar una cantidad importante de tiempo. Son cruciales las decisiones sobre entidades débiles y herencia.

Luego viene lo complicado: las dependencias funcionales. Es esencial una buena revisión de dependencias funcionales. Este es un problema no trivial: sale de su conocimiento del problema. En particular, la observación debería ir sobre las relaciones n-arias, con $n > 2$, sobre los atributos multivaluados compuestos y sobre aquellos atributos que ”casi” son derivados. Otra fuente de dependencias funcionales reside en las cardinalidades del diagrama E/R (ojo con las (1,1) ó (0,1)). Las relaciones con muchos atributos también pueden albergar bastantes dependencias funcionales (de entre los trabajos que corregí, hay varios de estos casos). Y obviamente están las relaciones de la forma llave $\to$ relación y las triviales[4].

Luego de obtenidas las dependencias funcionales, es menester recalcular las llaves. Conocer **todas** las llaves candidato es una obligación para determinar la forma normal de sus relaciones. Por supuesto, las llaves candidato son minimales.

El siguiente paso es obtener la forma normal de todas las relaciones del modelo. Especialmente críticas son las relaciones con muchos atributos, puesto que pueden albergar muchas dependencias funcionales. Revísese muy bien para buscar relaciones que no cumplan con FNBC. Además es importante saber si ya cumplen 3FN.

Toda relación en FNBC deberá ser normalizada, en lo posible, para obtener una descomposición FNBC preservando la información y las dependencias. Si esto no es posible, conformarse con lograr 3FN. Naturalmente, todo esto deberá estar muy bien explicado.

Y no se confíen. Recuerdo haber visto entidades laaaargas que pueden tener bastantes dependencias funcionales, relaciones ternarias y cuaternarias que ciertamente albergaban dependencias funcionales (y que llevarían a reducir la llave), y muchas restricciones relacionadas con las cardinalidades. Si bien no recuerdo haber visto modelos sobre conocimiento técnico, aún es posible descubrir dependencias sobre temas más cercanos a la cultura general.

[4] Omita las dependencias triviales