UNLAM - Base de datos

Normalización

Clase 2

Normalización

El proceso de normalización fue introducido por Codd en el año 1974. Allí se habían creado la primera forma normal (1FN), segunda forma normal (2FN) y tercera forma normal (3FN). Posteriormente Boyce y Codd crearon la forma normal de Boyce Codd (FNBC), haciendo más restrictiva la normalización.

Normalización

- *Menos espacio de almacenamiento*
- *Actualizaciones más rápidas*
- *Menor inconsistencia de datos*
- *Relaciones más claras*
- *Procedimientos más sencillos para la inserción de datos*
- *Estructura de datos flexible*

Un mal diseño puede ocasionar:

- *Redundancia*
- *Anomalias (inserción, actualización, eliminación)*

Normalización

El proceso de la normalización se basa en la descomposición del esquema, utilizando las dependencias funcionales. Existen varias formas normales. Podemos adecuar nuestro modelo a cada una de ellas, sabiendo que a medida que aumentamos la forma normal, estaremos agregando más restricciones a nuestro modelo.

Pérdida de dependencias funcionales

Sea un esquema de relación R con un conjunto de dependencias funcionales F y una descomposición D = { R1, R2, …, Rm} de R. Sea Fi el conjunto de todas las dependencias funcionales no triviales de F^+ que mencionan solamente atributos del subesquema Ri de la descomposición.

Decimos que D es una descomposición <u>sin Pérdida de Dependencias Funcionales</u> si se verifica que:

$$F \equiv F_1 \cup F_2 \cup \ldots \cup F_m$$

Recordar

¿Son F y G equivalentes?

$$F \stackrel{?}{\equiv} G$$

$$\downarrow$$

$$F^+ \stackrel{?}{=} G^+$$

$F \stackrel{?}{\subseteq} G^+$	$G \stackrel{?}{\subseteq} F^+$
$A^+_G = A\mathbf{BC}$	$D^+_F = D\mathbf{ABC}$
$B^+_G = B\mathbf{A}C$	$B^+_F = B\mathbf{AC}$
$D^+_G = D\mathbf{A}BC$	$A^+_F = A\mathbf{B}C$

Pérdida de dependencias funcionales

Ejemplo 1

Supongamos el esquema R (ABCDEF) y el conjunto F= {A→C, BC→A, D→F, AE→F, DC→E}.
Luego, supongamos que R se descompone en los siguientes 4 subesquemas:

R_1 (AC)
R_2 (DF)
R_3 (ADE)
R_4 (ABD)

Cada subesquema tendrá su propio conjunto de dependencias funcionales:

$F_1 = \{A \rightarrow C\}$
$F_2 = \{D \rightarrow F\}$
$F_3 = \{AD \rightarrow E\}$

Ya que:
$A^+_F = \{AC\}$
$D^+_F = \{DF\}$
$E^+_F = \{E\}$
$AD^+_F = \{ADCF\mathbf{E}\}$
$AE^+_F = \{AECF\}$
$DE^+_F = \{DEF\}$

$F_4 = \{\ \}$

Ya que:
$A^+_F = \{AC\}$
$B^+_F = \{B\}$
$D^+_F = \{DF\}$
$AB^+_F = \{ABC\}$
$AD^+_F = \{ADCFE\}$
$BD^+_F = \{BDF\}$

Pérdida de dependencias funcionales

Ejemplo

Ahora veremos si hubo pérdida de dependencias funcionales. Para ello, hay que verificar si el conjunto F original es equivalente a la unión de los cuatro conjuntos F_i de los subesquemas.

$$F \overset{?}{\equiv} F_1 \cup F_2 \cup F_3 \cup F_4$$

$$\{A \rightarrow C,\ BC \rightarrow A,\ D \rightarrow F,\ AE \rightarrow F,\ DC \rightarrow E\} \overset{?}{\equiv} \{A \rightarrow C\} \cup \{D \rightarrow F\} \cup \{AD \rightarrow E\} \cup \{\}$$

$$\{A \rightarrow C,\ BC \rightarrow A,\ D \rightarrow F,\ AE \rightarrow F,\ DC \rightarrow E\} \overset{?}{\equiv} \{A \rightarrow C,\ D \rightarrow F,\ AD \rightarrow E\}$$

Llamaremos G al conjunto resultante de la unión de los 4 conjuntos F_1, F_2, F_3 y F_4. Para verificar si F y G son equivalentes se debe verificar si F cubre a G y viceversa.
En este caso, a simple vista vemos que G no cubre a F, ya que por ejemplo la dependencia $BC \rightarrow A$ de F no existe en G_+.

$$BC^+ \text{ en } G = \{BC\}$$

Por lo tanto, podemos decir que en este caso la descomposición tiene Pérdida de Dependencias Funcionales.

Pérdida de Información

Sea un esquema de relación R con un conjunto de dependencias funcionales F y una descomposición D = {R1, R2, …, Rm} de R, decimos que D es una descomposición sin Pérdida de Información si, para cada instancia r de R que satisface las dependencias funcionales de F, se verifica que:

$$r = \pi_{R1}(r)\ |x|\ \pi_{R2}(r)\ |x|\ \ldots\ |x|\ \pi_{Rm}(r)$$

Es decir, si luego de la descomposición es posible reconstruir la relación inicial r mediante la junta natural de cada subesquema Ri, entonces decimos que no hubo pérdida de información.
Si al hacer la junta, se pierden algunas tuplas o bien se obtienen tuplas adicionales con información errónea (tuplas espurias), entonces decimos que sí hubo pérdida de información.

También puede ocurrir que al hacer la junta natural de los subesquemas, la relación resultante tenga menos atributos que el r original, en ese caso también decimos que hubo pérdida de información.

Pérdida de Información
Ejemplo

Supongamos el esquema de relación R(Patente, Modelo, Color, Marca, Motor, Combustible) y la siguiente instancia "r" de dicho esquema:

r

Patente	Modelo	Color	Marca	Motor	Combustible
IKU-496	206 Generation	Rojo	Peugeot	1.4	Nafta
HRV-709	207 Compact	Blanco	Peugeot	1.6	Nafta
GVR-286	Vectra GT	Negro	Chevrolet	2.4	Nafta
ILP-456	207 Compact	Azul	Peugeot	1.8	Diesel
HBN-142	206 Generation	Gris	Peugeot	1.4	Nafta

Evaluaremos si la siguiente descomposición de R es con o sin pérdida de información:
R1(Patente, Modelo, Color) R2(Modelo, Marca) R3(Modelo, Motor, Combustible)

Según la definición, no habrá P.I. si se cumple: $r = \pi_{R1}(r)\ |x|\ \pi_{R2}(r)\ |x|\ \pi_{R3}(r)$

$\pi_{R1}(r)$

Patente	Modelo	Color
IKU-496	206 Generation	Rojo
HRV-709	207 Compact	Blanco
GVR-286	Vectra GT	Negro
ILP-456	207 Compact	Azul
HBN-142	206 Generation	Gris

$\pi_{R2}(r)$

Modelo	Marca
206 Generation	Peugeot
207 Compact	Peugeot
Vectra GT	Chevrolet

$\pi_{R3}(r)$

Modelo	Motor	Combustible
206 Generation	1.4	Nafta
207 Compact	1.6	Nafta
Vectra GT	2.4	Nafta
207 Compact	1.8	Diesel

Pérdida de Información

Ejemplo

π_{R1} (r)

Patente	Modelo	Color
IKU-496	206 Generation	Rojo
HRV-709	207 Compact	Blanco
GVR-286	Vectra GT	Negro
ILP-456	207 Compact	Azul
HBN-142	206 Generation	Gris

π_{R2} (r)

Modelo	Marca
206 Generation	Peugeot
207 Compact	Peugeot
Vectra GT	Chevrolet

π_{R3} (r)

Modelo	Motor	Combustible
206 Generation	1.4	Nafta
207 Compact	1.6	Nafta
Vectra GT	2.4	Nafta
207 Compact	1.8	Diesel

π_{R1} (r) |x| π_{R2} (r)

Patente	Modelo	Color	Marca
IKU-496	206 Generation	Rojo	Peugeot
HRV-709	207 Compact	Blanco	Peugeot
GVR-286	Vectra GT	Negro	Chevrolet
ILP-456	207 Compact	Azul	Peugeot
HBN-142	206 Generation	Gris	Peugeot

π_{R1} (r) |x| π_{R2} (r) |x| π_{R3} (r)

Patente	Modelo	Color	Marca	Motor	Combustible
IKU-496	206 Generation	Rojo	Peugeot	1.4	Nafta
HRV-709	207 Compact	Blanco	Peugeot	1.6	Nafta
HRV-709	207 Compact	Blanco	Peugeot	1.8	Diesel
GVR-286	Vectra GT	Negro	Chevrolet	2.4	Nafta
ILP-456	207 Compact	Azul	Peugeot	1.6	Nafta
ILP-456	207 Compact	Azul	Peugeot	1.8	Diesel
HBN-142	206 Generation	Gris	Peugeot	1.4	Nafta

Pérdida de Información

Ejemplo

R Original

Patente	Modelo	Color	Marca	Motor	Combustible
IKU-496	206 Generation	Rojo	Peugeot	1.4	Nafta
HRV-709	207 Compact	Blanco	Peugeot	1.6	Nafta
GVR-286	Vectra GT	Negro	Chevrolet	2.4	Nafta
ILP-456	207 Compact	Azul	Peugeot	1.8	Diesel
HBN-142	206 Generation	Gris	Peugeot	1.4	Nafta

La junta natural de las tres tablas anteriores da como resultado:

π_{R1} (r) |x| π_{R2} (r) |x| π_{R3} (r)

Patente	Modelo	Color	Marca	Motor	Combustible	
IKU-496	206 Generation	Rojo	Peugeot	1.4	Nafta	
HRV-709	207 Compact	Blanco	Peugeot	1.6	Nafta	
HRV-709	**207 Compact**	**Blanco**	**Peugeot**	**1.8**	**Diesel**	Tupla espuria
GVR-286	Vectra GT	Negro	Chevrolet	2.4	Nafta	
ILP-456	**207 Compact**	**Azul**	**Peugeot**	**1.4**	**Nafta**	Tupla espuria
ILP-456	207 Compact	Azul	Peugeot	1.8	Diesel	
HBN-142	206 Generation	Gris	Peugeot	1.4	Nafta	

Finalmente podemos ver que no se cumple r = π_{R1} (r) |x| π_{R2} (r) |x| π_{R3} (r)
Por lo tanto la descomposición es con PERDIDA DE INFORMACIÓN.

Normalización

Conceptos

- ***Atributo atómico***: *Sólo puede contener un único valor en su instancia.*

- ***Dependencia parcial***:
  R1 (A,B,C,D) F1 = {A→B; C→D; AC→ D} CC={AC}
  A→B *es una dependencia parcial porque B depende parcialmente de la clave candidata.*

- ***Dependencia transitiva***:
  R2 (A,B,C,D) F2 = {A→B; B→C; C→D; A→C} CC={A}
  A→C *es una dependencia transitiva porque se puede obtener aplicando el axioma de transitividad entre A→B y B→C.*

Atributo Primo

Atributo que forma parte de una CC compuesta. Ej: CC={ABC}, los atributos A, B y C son llamados atributos primos.

Formas Normales *

Modelo Sin Normalizar

1 FN → *Sólo atributos atómicos*

2 FN → *Sin dependencias parciales*

3 FN → *Sin dependencias Transitivas**

FNBC → *Dependencia total en la clave*

* Una dependencia functional $X \rightarrow Y$ en un esquema de relación R es una **dependencia transitiva** si existe un conjunto de atributos Z que ni es clave candidata ni un subconjunto de ninguna clave de R, y se cumple tanto $X \rightarrow Z$ como $Z \rightarrow Y$.

Formas Normales

Para toda Dependencia Funcional de R, tal que X → Y:

FNBC

X es superclave

3 FN

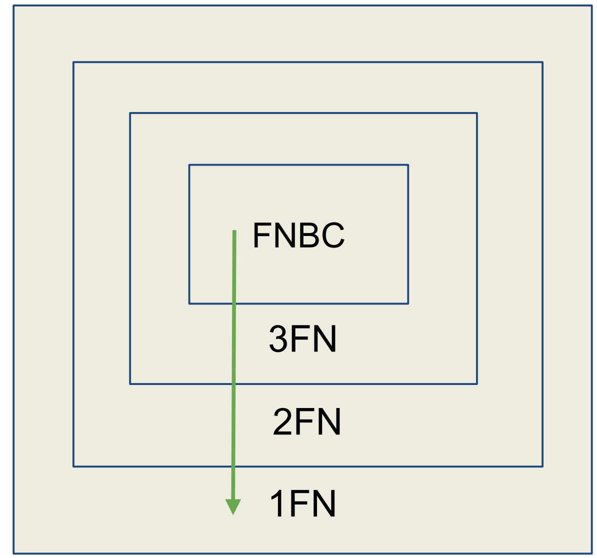
X es superclave ó
Y es primo

2 FN

X es superclave ó
Y es primo ó
X no es un subconjunto de alguna de las CC

1 FN

No deben existir atributos multivaluados



Algoritmos de Descomposición

Existen dos algoritmos que nos permiten realizar la descomposición de una relación:

- *Algoritmo de 3FN:*
  - *No existe pérdida de dependencias funcionales*
  - *No existe pérdida de información*
- *Algoritmo de FNBC:*
  - *Puede o no existir pérdida de dependencias funcionales*
  - *No existe pérdida de información*

Algoritmo de 3FN

Pasos:

1) Calcular Fmin

2) Para cada miembro izquierdo X que aparezca en Fmin, crear un esquema de relación {X unión A1 unión A2 unión An} donde X→A1, X→A2, … , X→An sean todas DF de Fmin.

3) Si ninguno de los esquemas resultantes contiene una clave candidata de R, crear un esquema adicional que contenga una clave candidata de R.

Veamos un ejemplo:
Dado R (ABCDEFG) y F= {A→BC, AD→G, AC→E, C→A, B→C, F→B}

Primero calculamos el Fmin:

Fmin ={A→B, AD→G, A→E, C→A, B→C, F→B}

Luego, analizamos en qué forma normal se encuentra R, para ello calculamos sus Claves Candidatas:

CC={DF}

Analizando las dependencias funcionales de Fmin, vemos que R se encuentra en 1FN (ya que F→B viola 2FN).

Finalmente, descomponemos R en subesquemas que cumplan 3FN:

R1(ABE)	R2(ADG)	R3(AC)	R4(BC)	R5(BF)	R6(DF)
F1={A→B, A→E}	F2={AD→G}	F3={C→A}	F4={B→C}	F5={F→B}	F6={}

Nota: El último esquema R6 se agrega porque ninguno de los subesquemas anteriores contiene la clave candidata DF.

Ejemplo Algoritmo de 3FN

R(A,B,C,D,E,F,G)
F={A →BC, AD→G, AC→E, C→A, B→C, F→B}

C.C. = {DF}

1 FN

1) Calcular Fmin

Fmin = {A →B, AD→G, A→E, C→A, B→C, F→B}

2)Para cada miembro izquierdo X que aparezca en Fmin…..

R1 (A,B,E)   F1={A →B, A→E}
R2 (A,D,G)   F2={AD →G}
R3 (A,C)   F3={C →A}
R4 (B,C)   F4={B →C}
R5 (B,F)   F5={F →B}

3)Si ninguno de los esquemas resultantes contiene una clave candidata de R, crear un esquema adicional que contenga una clave de R.

Agrega **R6 (D,F)   F6={}**

Algoritmo de 3FN

Ej 20

Algoritmo de FNBC

Dado el esquema R y el conj. de DF F (no es necesario trabajar con Fmin, pero si es aconsejable):
Tomar cualquier DF $X \rightarrow Y$ perteneciente a F que no sea trivial y que viole FNBC, y hacer 2 subesquemas:

R1=**(X U Y)** y R2=**(R - Y)**

Si alguno de los dos subesquemas obtenidos sigue violando FNBC, aplicar nuevamente el punto anterior sobre dicho esquema. Repetir ese paso, e ir dividiendo en subesquemas, tantas veces como sea necesario, hasta lograr obtener todos subesquemas que cumplan FNBC.

División de las Dependencias Funcionales:
Las dependencias de F deberán repartirse a F1 o F2 según corresponda (puede ser que alguna dependencia no pueda incluirse en ninguno de los dos subconjuntos y en ese caso la misma se perderá, pero esa pérdida puede salvarse intentando inferir alguna otra dependencia equivalente en base a ella, que si pueda ubicarse en F1 o F2).

Ejemplo de Algoritmo de FNBC

Dado R(MNOPQS) y F={M→ON, N→MO, O→NM, MQ→SO, OP→Q}

En primer lugar, calcularemos Fmin para simplificar el desarrollo:

F1={ M→O, M→N, N→M, N→O, O→N, O→M, MQ→S, MQ→O, OP→Q }
F2={ M→O, M→N, N→M, N→O, O→N, O→M, MQ→S, OP→Q}
F3={ M→N, N→O, O→M, MQ→S, OP→Q } = Fmin

¿En qué forma normal se encuentra R?
Para responder a esa pregunta debemos calcular las Claves Candidatas y luego evaluar una a una las dependencias funcionales.

CC = {PM, PN, PO}

M→N	cumple 3FN
N→O	cumple 3FN
O→M	cumple 3FN
MQ→S	cumple 2FN
OP→Q	cumple FNBC

Repuesta: R se encuentra en 2FN

Algoritmo de FNBC

{ M→N, N→O, O→M, MQ→S, OP→Q }

M→N

R1(MN)
F1={M→N, N→M}

DF inferida

R2(MOPQS)
F2={O→M, MQ→S, OP→Q, M→O}

DF inferida

CC={PM, PO}

O→M

R3(OM)
F3={O→M, M→O}

R4(OPQS)
F3={OP→Q, OQ→S }

DF inferida

CC={PO}

OQ→S

R5(OQS)
F5={OQ→S}

R6(OPQ)
F6={OP→Q}

La descomposición obtenida es:
R1, R3, R5 y R6 con F1, F3, F5 y F6

Inferencias

1) M->N (Dada)
2) N->O (Dada)
3) M->O (?)

1) N->O (Dada)
2) O->M (Dada)
3) N->M (?)

1) O->M (Dada)
2) MQ->S (Dada)
3) ....

Teorema de Health

- *Permite determinar si una descomposición posee o no pérdida de información, pero sólo se podrá utilizar si la relación posee dos subesquemas.*

$$\text{La DF } (R1 \cap R2) \rightarrow (R1 - R2) \in F+$$

ó

$$\text{La DF } (R1 \cap R2) \rightarrow (R2 - R1) \in F+$$

- ***Ejemplo***:

R(Y,Z,W,X) F={Y → Z, W → X}

R1 (Y, Z,W) F1={Y → Z} / R2(W,X) F2={W → X}

$(R1 \cap R2) \rightarrow (R1 - R2)$ $\quad W \rightarrow YZ \quad \notin \quad F+$

$(R1 \cap R2) \rightarrow (R2 - R1)$ $\quad W \rightarrow X \quad \in \quad F+$

→ **No existe pérdida de Información**

Conceptos Vistos

Pérdida de Dependencias Funcionales

Pérdida de Información

Algoritmos (3FN y FNBC)

Tableau

Teorema de Heath