Normalización

formas normales

Presentación

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.

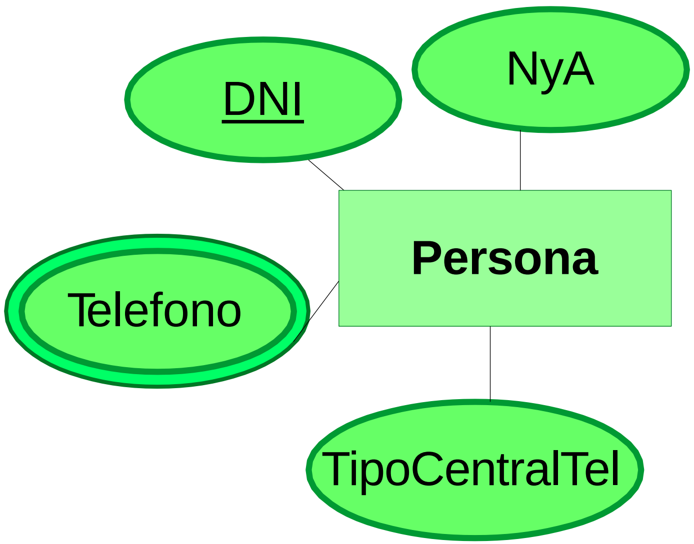
Beneficios de la normalización

- *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)*

Introducción



1. ¿Cómo lo pasamos al modelo relacional?
Persona (DNI, NyA, TipoCentralTel)
Telefono (DNI, Tel)

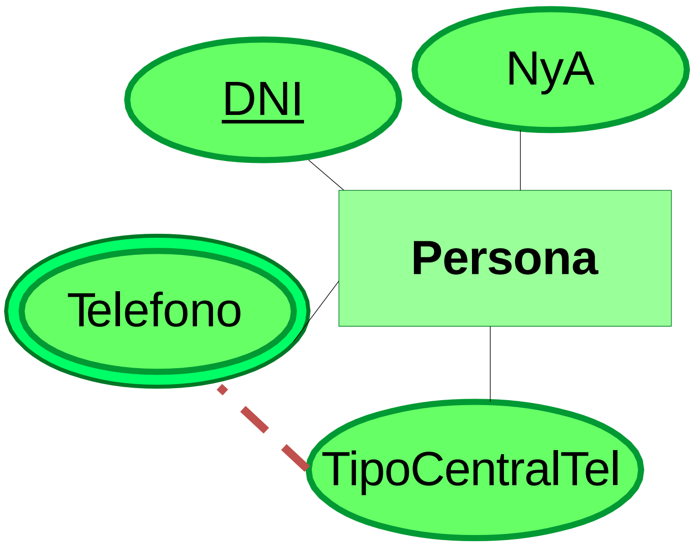
2. ¿Cuáles son sus dependencias funcionales?
Fpersona={DNI → NyA; DNI → TipoCentralTel}
Ftelefono={DNI,Tel → DNI,Tel}

3. ¿Qué sucede con la depenencia DNI → TipoCentralTel? ¿Es correcta? Ó debería ser que el Tel → TipoCentralTel?

4.¿Y si cambiamos el diseño?
Persona (DNI, NyA)   Fp={DNI → NyA}
Telefono (DNI, Tel, TipoCentralTel)
Ft={DNI,Tel → TipoCentralTel;
Tel → TipoCentralTel }

5.¿Qué sucede ahora si varias personas poseen el mismo número de teléfono?

Introducción



6. ¿Cómo lo cambiaríamos para que no genere información redundante?

~~*Persona (DNI, NyA) Fp={DNI → NyA}*~~
~~*Telefono (DNI, Tel, TipoCentralTel)*~~
~~*Ft={DNI,Tel → TipoCentralTel;*~~
~~*Tel → TipoCentralTel }*~~

Persona (DNI, NyA)
Fp={DNI → NyA}
Telefono (DNI, Tel)
Ft={DNI,Tel → DNI,Tel}
TipodeCentral (Tel, TipoCentralTel)
Ftc={Tel → TipoCentralTel}

7. ¿Qué método empleamos para realizar este cambio?

DESCOMPOSICIÓN

8. ¿En qué nos basamos para realizar la descomposición?

En las dependencias funcionales.

Conceptos relacionados

- ***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 A 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 agregar

Formas Normales

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.

Formas Normales

Modelo Sin Normalizar

1 FN → *Sólo atributos atómicos*

2 FN → *Sin dependencias parciales*

Sin dependencias Transitivas ← **3 FN**

Dependencia total en la clave ← **FNBC**

Formas Normales

Para toda Dependencia Funcional de R, tal que $X \rightarrow Y$:

1 FN	→ *Sólo atributos atómicos*	No deben existir atributos multivaluados
2 FN	→ *Sin dependencias parciales*	X es superclave ó Y es primo ó X no pertenece a ninguna clave
3 FN	→ *Sin dependencias Transitivas*	X es superclave ó Y es primo
FNBC	→ *Dependencia total en la clave*	X es superclave

Identificar Formas Normales

1	R1(A,B,C)	F1={A $\rightarrow$ B}	?
2	R2(A,B,C)	F2={AB $\rightarrow$ C, C $\rightarrow$ B}	?
3	R3(A,B,C)	F3={A $\rightarrow$ B, B $\rightarrow$ C}	?
4	R4(A,B)	F41={A $\rightarrow$ B} F42={B $\rightarrow$ A} F43={A $\rightarrow$ B, B $\rightarrow$ A} F44={}	?

Identificar Formas Normales

1	R1(A,B,C)	$F1=\{A \rightarrow B\}$	CC= ? FN= ?
2	R2(A,B,C)	$F2=\{AB \rightarrow C, C \rightarrow B\}$	CC= ? FN= ?
3	R3(A,B,C)	$F3=\{A \rightarrow B, B \rightarrow C\}$	CC= ? FN= ?
4	R4(A,B)	$F41=\{A \rightarrow B\}$	CC= ? FN= ?
		$F42=\{B \rightarrow A\}$	CC= ? FN= ?
		$F43=\{A \rightarrow B, B \rightarrow A\}$	CC= ? FN= ?
		$F44=\{\}$	CC= ? FN= ?

Descomposición

La descomposición de un modelo relacional, puede provocar:

- *Pérdida de Dependencias funcionales*
- *Pérdida de Información*

Pérdida de dependencias funcionales

Se dice que una descomposición conserva todas sus dependencias funcionales si:

$$Fd=\{ F1 \cup F2 \cup F3... \cup Fi\}^+ = F^+$$

Es decir, si se pueden obtener todas las dependencias funcionales originales, del conjunto de dependencias funcionales de la descomposición.

Pérdida de dependencias funcionales

Algoritmo:

$x \rightarrow y$

$r = x$

while (existan cambios en r) do

for each Ri in (descomposición)

$t=(r \cap Ri)^+ \cap Ri$

$r = r \cup t$

Si “r” posee todos los atributos “y”, entonces la dependencia se ha conservado

Pérdida de dependencias funcionales

Ejemplo 1 A

R (A, B, C, D)

F = {A → B, B → C, C → D)

Descomposición:

R1 (A, B) F1={A → B}

R2 (B, C) F2={B → C}

R3 (C, D) F3={C → D}

¿Se perdieron dependencias funcionales en la descomposición?

Fd={ F1 U F2 U F3... U Fi}+ = F+

Pérdida de dependencias funcionales

Ejemplo 1 B

R (A, B, C, D)

F = {A → B, B → C, C → D, A → C, B → D)

Descomposición:

R1 (A, B) F1={A → B}
R2 (B, C) F2={B → C}
R3 (C, D) F3={C → D}

¿Se perdieron dependencias funcionales en la descomposición?

Fd={ F1 U F2 U F3... U Fi}+ = F+

Pérdida de dependencias funcionales

En la verificación de las Dfs, podrían suceder dos casos:

1)df $\in$ (F1 U F2 U ...Fi): que la dependencia funcional f se encuentre en las UFi. Si esto sucede, ya sabemos que esa dependencia funcional, no se ha perdido.

2)df $\notin$ (F1 U F2 U ...Fi): que la depenencia funcional de F no se encuentre en la UFi . Entonces podría ser obtenida si utilizamos axiomas o reglas y si lo logramos, sería una dependencia funcional derivada. Con lo cual, tampoco se habría perdido. Si tampoco se puede derivar, concluímos que esa df se ha perdido.

Pérdida de dependencias funcionales

Ejemplo 2

Veamos algunos ejemplos:

R (DNI, NyA, Domicilio, Mail)
F={DNI → NyA, Domicilio}

Podríamos descomponer en:

R1 (DNI,NyA, Domicilio) F1={DNI → NyA, Domicilio}
R2 (DNI,Mail) F2={ }

(F1 U F2)+ = F+ → Se preservaron las dependencias

Pérdida de dependencias funcionales

R (DNI, NyA, Empresa, MailCorporativo)
F={DNI→NyA,Empresa; Empresa→MailCorporativo;
DNI→MailCoporativo}

Podríamos descomponer en:
R1 (DNI,NyA, Empresa) F1={DNI → NyA, Empresa}
R2 (Empresa,MailCoporativo) F2={ Empresa → MailCoporativo}

(F1 U F2)= {DNI→NyA,Empresa; Empresa→MailCoporativo}

¿Falta DNI → MailCoporativo? A través de la transitividad la dependencia se podría obtener. Por lo cual, también se conservan las dependencias funcionales. Se cumple que:

{F1 U F2) + = F+

DNI+(F1UF2)={DNI,NyA,Empresa,MailCoporativo}
DNI+(F)={DNI,NyA,Empresa,MailCoporativo}

Pérdida de dependencias funcionales

R (A, B, C}
F={A→B; B→C; C→B; A→C}

Podríamos descomponer en:
R1(A,C)   F1={A→C}
R2(B,C)   F2={B→C; C→B}

(F1 U F2) ={A→C; B→C; C→B}

¿Falta A→B? ¿Se perdió?
No, porque esas dependencias se pueden derivar de las dependencias de F1 y F2.
A+(F1UF2)={A,C,B}   */ A→B? A→C y C→B => A→B (x transitiv.)*
A+(F)={A,C,B}

Pérdida de dependencias funcionales

Dado:

R (A, B, C)

F={A→C; BC→A}

Podríamos descomponer en:

R1(A,C)   F1={A→C}

R2(A,B)   F2={}

(F1 U F2) ={A→C}

¿Falta BC→A? ¿Se perdieron Dfs?

Sí, porque esas dependencias No se pueden derivar de las dependencias de F1 y F2.

BC+(F1UF2)={B,C}

BC+(F)={B,C,A}

Pérdida de Información

Se dice que una descomposición posee pérdida de información, cuando no se puede reconstruir la tupla original a partir de la tupla de la cada R_i

$$r = r1 \;|X|\; r2 \;|X|\; r3 \;\ldots\; |X|\; r_i \qquad / \qquad r_i = \pi_{R_i}(r)$$

En toda descomposición, cada tupla debería poder reconstruirse. Si no pudiera reconstruirse se dice que esa descomposición es inválida.

Pérdida de Información

Veamos algunos ejemplos:
R (DNI, NyA, Empresa, MailCorp)
F={DNI→NyA,Empresa; Empresa→MailCorp}

Podríamos descomponer en:
R1 (DNI,NyA,Empresa) F1={DNI→NyA,Empresa}
R2 (Empresa,MailCorp) F2={Empresa→ MailCorp }

R

DNI	NyA	Empresa	MailCorp
1234	Juan	E1 S.A.	e1@e1
9999	Ana	E2 S.A.	e2@e2
5656	Lola	E1 S.A.	e1@e1

DNI	NyA	Empresa
1234	Juan	E1 S.A.
9999	Ana	E2 S.A.
5656	Lola	E1 S.A.

Empresa	MailCorp
E1 S.A.	e1@e1
E2 S.A.	e2@e2

R (1234, Juan, E1 S.A, e1@e1) **=** R1 (1234, Juan, E1 S.A) |X| R2 (E1 S.A., e1@e1)

No hubo pérdida de Información

Pérdida de Información

Veamos algunos ejemplos:
R(DNI, NyA, Empresa, MailCorp)
F={DNI→NyA,Empresa; Empresa→MailCorp}

Podríamos descomponer en:
R1 (DNI,NyA,Empresa) F1={DNI→NyA,Empresa}
R2 (MailCorp) F2={ }

DNI	NyA	Empresa	MailCorp
1234	Juan	E1 S.A.	e1@e1
9999	Ana	E2 S.A.	e2@e2
5656	Lola	E1 S.A.	e1@e1

DNI	NyA	Empresa
1234	Juan	E1 S.A.
9999	Ana	E2 S.A.
5656	Lola	E1 S.A.

?

MailCorp
e1@e1
e2@e2

R (1234, Juan, E1 S.A, e1@e1) **<>** R1 (1234, Juan, E1 S.A) |X| R2 (e1@e1) = { }

Hubo pérdida de Información

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

1) Calcular Fmin

2)Para cada miembro izquierdo X que aparezca en Fmin, crear un esquema de relación (XUA1UA2....UAn) donde X → A1, X → A2, .. X → An) sean todas Df de Fmin

3)Colocar los atributos restantes, no incluidos en paso 2, en un sólo esquema de relación

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

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, crear un esquema de relación (XUA1UA2....UAn) donde X → A1, X → A2, .. X → An) sean todas Df de 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) Colocar los atributos restantes, no incluidos en paso 2, en un sólo esquema de relación

No agrega ningún esquema

4)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={}**

Ejemplo Algoritmo de 3FN

R(C,P,D,I,K,N)
F={PC→D, PC→I, DI→K, K→I}

C.C. = {PCN}

1 FN

1) Calcular Fmin

Fmin = {PC→D, PC→I, DI→K, K→I}

2)Para cada miembro izquierdo X que aparezca en Fmin, crear un esquema de relación (XUA1UA2....UAn) donde X → A1, X → A2, .. X → An) sean todas Df de Fmin

R1 (P,C,D,I)	**F1={PC→D, PC→I}**	**CC1={PC}**
R2 (D,I,K)	**F2={DI→K, K→I}**	**CC2={DI,DK}**

3) Colocar los atributos restantes, no incluidos en paso 2, en un sólo esquema de relación

Agrega **R3 (N)** **F3={}** **CC3={N}**

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

Agrega **R4 (P,C,N)** **F4={}** **CC4={PCN}**

Algoritmo de FNBC

1)Para cada DF X → Y en F, que no sea trivial y que viole las condiciones de FNBC, hacer 2 esquemas:

R1=(X U Y) y R2 (R – Y)

Para alguna de las dependencias funcionales de F, no pudiera inclurise en alguno de los subesquemas, esa Dfs podría perderse. Para evitar la pérdida de Dfs se podría inferir alguna Df equivalente que pueda ir en F1 ó F2.

2)Si los subesquemas resultantes siguen sin cumplir las condiciones de FNBC, volver al punto 1.

Ejemplo de Algoritmo de FNBC

R(M,N,O,P,Q,S)
F={M →ON, N →MO, O →MN, MQ →SO, OP →Q}

C.C.={PM, PN, PO}
2 FN

Si bien no es obligatorio, se calcula el Fmin para comenzar por el menor F posible:

Fmin={M →N, N →O, O → M, MQ→S, OP →Q}

1) Para cada DF X → Y en F, que no sea trivial y que viole las condiciones de FNBC, hacer 2 esquemas: R1=(X U Y) y R2 (R – Y)

Fmin={<u>M →N</u>, N →O~~,~~ O → M, MQ→S, OP →Q}

R1(M,N)
F1={M→N}
CC1={M}

R2(M,O,P,Q,S)
F2={<u>O→M</u>, MQ→S, OP→Q}
CC2={PO}

R3(O,M)
F3={O→M}
CC3={O}

R4(O,P,Q,S)
F4={OP→Q, OQ→S}
CC4={PO}

R5(O,Q,S)
F5={OQ→S}
CC5={OQ}

R6(O,P,Q)
F6={OP→Q}
CC6={OP}

R1 U R3 U R5 U R6

Algoritmo de pérdida de Información

- *Método Tableau*

  N relaciones

- *Teorema de Health*

  2 relaciones

Método Tableau

Dado un esquema R(A1,A2,..An) y D={R1, R2 .. Rn}, armar una grilla con lo siguiente:

1) Generar una columna por cada atributo de R
2) Generar una fila por cada relación de la descomposición Ri
3) En cada celda completar con aj o bij, dependiendo:

   a_j si $A \in R_i$

   b_{ij} si $A \notin R_i$

4)Para cada df $X \rightarrow A$ de F, verificar si se cumple en la grilla. Si se cumple, se deben cambiar los valores de a_j ó b_{ij}.
5)El procedimiento finalizará cuando se haya iterado y no haya cambios en el tableau.
6)Si encontramos una fila que posea todas "a_i", esto significará que no se perdió información. De lo contrario, la descomposición sería incorrecta.

Método Tableau: Ejemplo

R(A,B,C,D,E) / F={C→E, E→D, B→E,AD→B}
R1(A,B,C) R2 (C,D) R3 (A,D,E)

Tableau:

***Paso1**: Armar la grilla*

	A	B	C	D	E
R1	a1	a2	a3	b14	b15
R2	b21	b22	a3	a4	b25
R3	a1	b32	b33	a4	a5

***Paso 2**: Iterar por las dfs*

C → E

	A	B	C	D	E
R1	a1	a2	a3	b14	b15
R2	b21	b22	a3	a4	~~b25~~ **b15**
R3	a1	b32	b33	a4	a5

Método Tableau: Ejemplo

R(A,B,C,D,E) / F={C→E, E→D, AD→B, B→E}
R1(A,B,C) R2 (C,D) R3 (A,D,E)

Paso 2: *Iterar por las dfs*

E → D

	A	B	C	D	E
R1	a1	a2	a3	~~b14~~ **a4**	b15
R2	b21	b22	a3	a4	~~b25~~ **b15**
R3	a1	b32	b33	a4	a5

AD → B

	A	B	C	D	E
R1	a1	a2	a3	~~b14~~ **a4**	b15
R2	b21	b22	a3	a4	~~b25~~ **b15**
R3	a1	~~b32~~ **a2**	b33	a4	a5

B → E

	A	B	C	D	E
R1	a1	a2	a3	~~b14~~ **a4**	~~b15~~ **a5**
R2	b21	b22	a3	a4	~~b25~~ **b15**
R3	a1	~~b32~~ **a2**	b33	a4	a5

Método Tableau: Ejemplo

- *R(A,B,C) F={AC → B, B → C}*

FNBC

R'(A,B,C) F'={AC → B, B → C}

R1(B,C)
F1=(B → C}

R2(A,B)
F2={}

***Tableau*:**

Paso1: *Armar la grilla*

	A	B	C
R1	b11	a2	a3
R2	a2	a2	b23

Paso 2: *Iterar por las dfs F1 U F2*

	A	B	C
R1	b11	a2	a3
R2	a2	a2	~~b23~~ **a3**

Como se encontró una fila con todas a, se dice que NO hay pérdida de Información

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.*

$$La\ DF\ (R1 \cap R2) \rightarrow (R1 - R2) \in F+ \qquad ó$$
$$La\ DF\ (R1 \cap R2) \rightarrow (R2 - R1) \in F+$$

- ***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

- *Beneficios de la Normalización. Descomposición.*
- *Formas Normales*
  - *1 FN*
  - *2 FN*
  - *3FN*
  - *FNBC*
- *Pérdida de Dependencias Funcionales*
- *Pérdida de Información*
  - *Teorema de Health*
  - *Tableau*

Consultas

Gracias