

Unidad Didáctica 4: Diseño de Bases de Datos Relacionales

Parte 3: Diseño Lógico

U.D. 4.3

UD 4.3. Diseño lógico

1. Introducción

2. Transformación de las clases

2.1. Clases fuertes

2.2. Clases débiles

2.3. Clases especializadas

3. Transformación de las asociaciones

3.1. No reflexivas

3.2. Reflexivas

3.3. Atributos de enlace

3.4. Transformación de la asociación cuando se asocia con clases asociación

3.5. Elección de las directrices de las claves ajenas

4. Ejemplo

5. Teoría de la normalización

1. Diseño lógico

1ª FASE: Análisis

Investigación

Req. de
información

Req. de
procesos

2ª FASE: Diseño

Modelo semántico

Diseño conceptual

Esquema conceptual

Estática

Dinámica

Tecnología de gestión
de datos

Diseño lógico

Esquema
lógico

Esquemas de
transacciones

SGBD

Diseño físico

Esquema
físico

Diseño y
desarrollo de
Programas

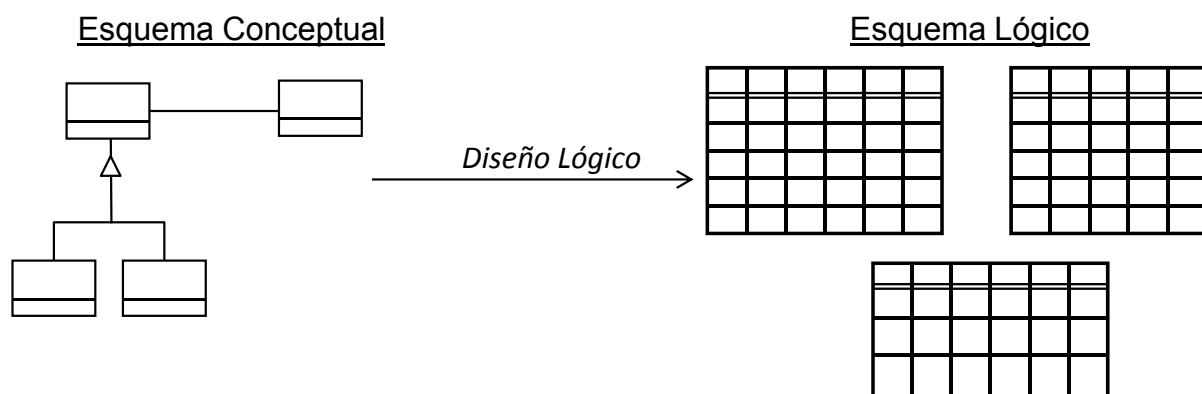
3ª FASE: Implantación

Carga de la
base de datos

3

1. Diseño lógico

Diseño lógico: transformación del esquema conceptual, que se encuentra descrito con un cierto modelo de datos, en estructuras descritas en términos del modelo de datos en el cual se base el sistema de gestión de bases de datos que se vaya a utilizar.



4

1. Diseño lógico

- Transformaciones: se basan en la definición de: CP, CAj, VNN, Único
- Aquellas propiedades expresadas en el diagrama que no se puedan representar en el esquema relacional deberán ser incluidas en una *lista de restricciones de integridad* para que sean controladas desde restricciones generales (*assertions*, *triggers* o programa).
- Cuando haya varios esquemas relacionales posibles:
 1. *Elegir el esquema con menos restricciones de integridad añadidas.*
 2. *Ante igualdad de restricciones, elegir el esquema con menos relaciones.*

5

UD 4.3 Diseño lógico

1. Introducción

2. Transformación de las clases

- 2.1. Clases fuertes
- 2.2. Clases débiles
- 2.3. Clases especializadas

3. Transformación de las asociaciones

- 3.1. No reflexivas
- 3.2. Reflexivas
- 3.3. Atributos de enlace
- 3.4. Transformación de la asociación cuando se asocia con clases asociación
- 3.5. Elección de las directrices de las claves ajenas

4. Ejemplo

5. Teoría de la normalización

6

2.1. Clases fuertes

Aparecen en A(...)
los que tienen
{x..1} e id, único

A
$a_0: \{id\}: t_{a_0}$ $a_1: \{\text{único}_1\}: \{0..1\}: t_{a_1}$ $a_2: \{1..1\}: t_{a_2}$ $a_3: \{0..1\}: t_{a_3}$ $a_4: \{1..*\}: t_{a_4}$ $a_5: \{0..*\}: t_{a_5}$ $a_6: \{0..1\}: t_{a_6}$ $a_{61}: t_{a_{61}}$ $a_{62}: t_{a_{62}}$

$A(a_0:t_{a_0}, a_1:t_{a_1}, a_2:t_{a_2}, a_3:t_{a_3}, a_{61}:t_{a_{61}}, a_{62}:t_{a_{62}})$

CP: { a_0 } -> id

Único: { a_1 }

VNN: { a_2 } -> {1..1}

$A4(a_0:t_{a_0}, a_4:t_{a_4}) \rightarrow \{0..*\}$

CP: { a_0, a_4 }

$A_4: \{1..*\}$

CAj: { a_0 } → A(a_0)

RI1: Todo valor que aparece en el atributo a_0 de A debe aparecer en el atributo a_0 de A4.

$A5(a_0:t_{a_0}, a_5:t_{a_5})$

CP: { a_0, a_5 }

CAj: { a_0 } → A(a_0) -> {0..*}

7

2.1. Clases fuertes

Persona
DNI: {id}: char NSS: {único ₁ }: {1..1}: char nombre: {1..1}: propio: {1..1}: char apellidos: {1..1}: char edad: {0..1}: int teléfonos: {0..*}: char

Persona(DNI:char, NSS:char, nombre_propio:char,
apellidos:char, edad:int,)
CP: {DNI}
Único: {NSS}
VNN: {NSS, nombre_propio, apellidos}

Contacto(DNI: char, teléfono:char)
CP: {DNI, teléfono}
CAj: {DNI} → Persona(DNI)

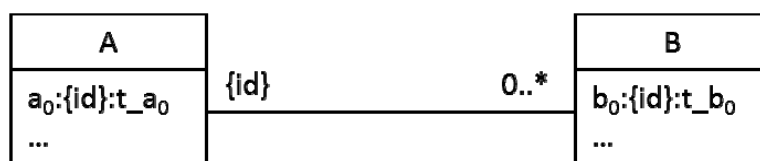
8

UD 4.3 Diseño lógico

1. Introducción
2. Transformación de las clases
 - 2.1. Clases fuertes
 - 2.2. Clases débiles
 - 2.3. Clases especializadas
3. Transformación de las asociaciones
 - 3.1. No reflexivas
 - 3.2. Reflexivas
 - 3.3. Atributos de enlace
 - 3.4. Transformación de la asociación cuando se asocia con clases asociación
 - 3.5. Elección de las directrices de las claves ajenas
4. Ejemplo
5. Teoría de la normalización

9

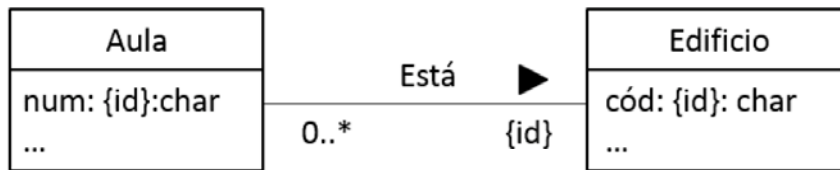
2.2. Clases débiles



$A(a_0:t_{a_0}, \dots)$
 $CP: \{a_0\}$

$B(b_0:t_{b_0}, a_0:t_{a_0}, \dots)$
 $CP: \{a_0, b_0\}$
 $CA^j: \{a_0\} \rightarrow A(a_0)$

2.2. Clases débiles

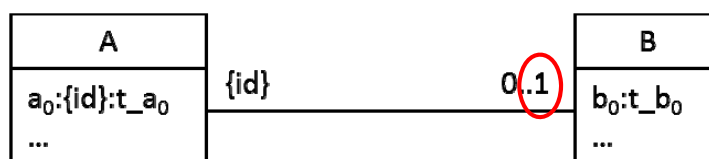


```
Edificio(cod:char,...)  
  CP:{cod}  
  Único:{NSS}
```

```
Aula(cod: char, num:char)  
  CP:{cod, num}  
  CAj:{cod}→ Edificio(cod)
```

11

2.2. Clases débiles

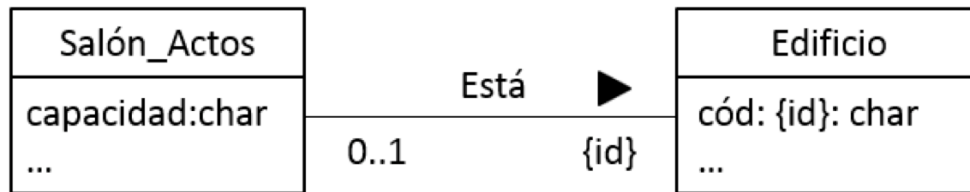


```
A(a0:ta0,...)  
  CP:{a0}
```

```
B(b0:tb0,a0:ta0,...)  
  CP:{a0}  
  CAj:{a0}→A(a0)
```

12

2.2. Clases débiles

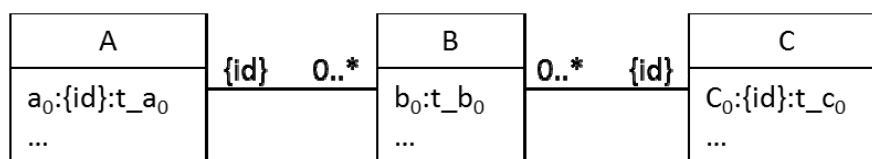


Edificio(cod:char,...)
 CP:{cod}
 Único:{NSS}

Salón_Actos(cod: char, capacidad:char, ...)
 CP:{cod}
 CAj:{cod}→ Edificio(cod)

13

2.2. Clases débiles



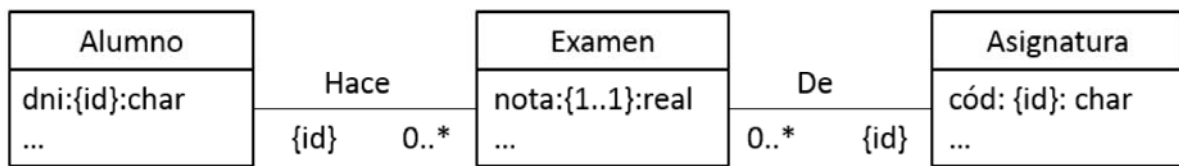
A(a₀:t_{a₀},...)
 CP:{a₀}

C(c₀:t_{c₀},...)
 CP:{c₀}

B(a₀:t_{a₀},c₀:t_{c₀},b₀:t_{b₀},...)
 CP:{a₀,c₀}
 CAj:{a₀}→A(a₀)
 CAj:{c₀}→C(c₀)

14

2.2. Clases débiles



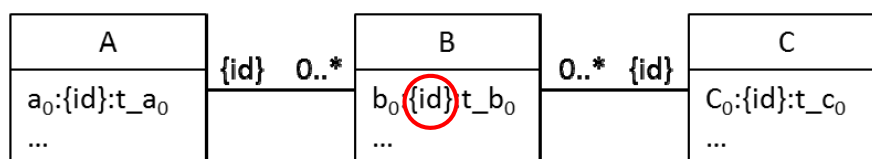
Alumno(dni:char,...)
 CP:{dni}

Asignatura(cod: char, ...)
 CP:{cod}

Examen(dni: char, cod:char, nota: real, ...)
 CP:{dni,cod}
 CAj:{dni}→ Alumno(dni)
 CAj:{cod}→ Asignatura(cod)
 VNN:{nota}

15

2.2. Clases débiles



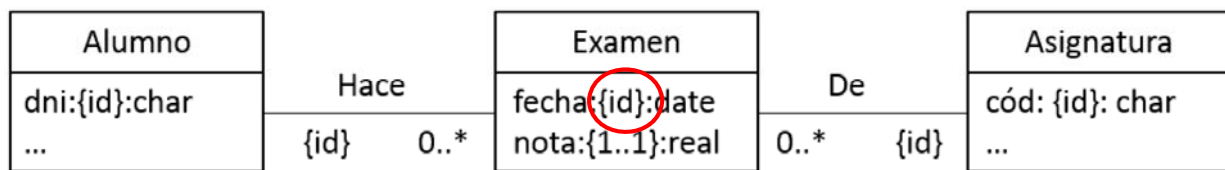
A(a₀:t_{a₀},...)
 CP:{a₀}

C(c₀:t_{c₀},...)
 CP:{c₀}

B(a₀:t_{a₀},c₀:t_{c₀},b₀:t_{b₀},...)
 CP:{a₀,c₀,b₀}
 CAj:{a₀}→A(a₀)
 CAj:{c₀}→C(c₀)

16

2.2. Clases débiles



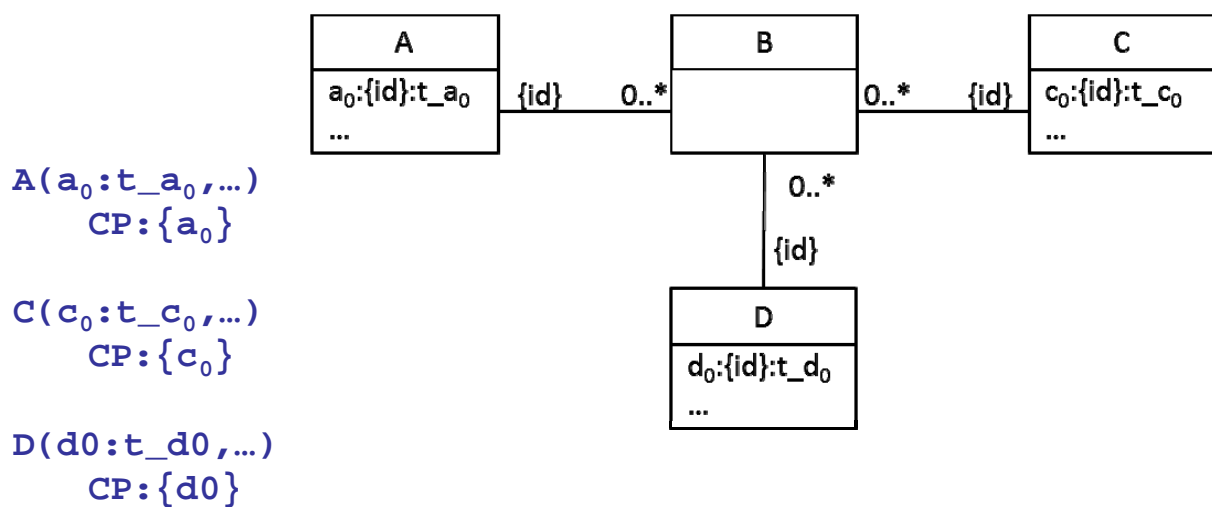
Alumno(dni:char,...)
CP:{dni}

Asignatura(cod: char, ...)
CP:{cod}

Examen(dni: char, cod:char, fecha: date, nota: real, ...)
CP:{dni, cod, fecha}
CAj:{dni}→ Alumno(dni)
CAj:{cod}→ Asignatura(cod)
VNN:{nota}

17

2.2. Clases débiles



A(a₀:t_{a0},...)
CP:{a₀}

C(c₀:t_{c0},...)
CP:{c₀}

D(d₀:t_{d0},...)
CP:{d₀}

B(a₀:t_{a0},c₀:t_{c0},d₀:t_{d0})
CP:{a₀,c₀,d₀}
CAj:{a₀}→A(a₀)
CAj:{c₀}→C(c₀)
CAj:{d₀}→D(d₀)

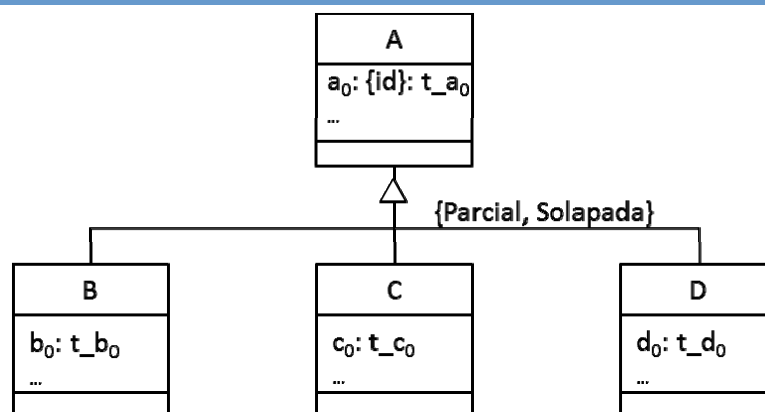
18

UD 4.3 Diseño lógico

1. Introducción
2. Transformación de las clases
 - 2.1. Clases fuertes
 - 2.2. Clases débiles
 - 2.3. Clases especializadas
3. Transformación de las asociaciones
 - 3.1. No reflexivas
 - 3.2. Reflexivas
 - 3.3. Atributos de enlace
 - 3.4. Transformación de la asociación cuando se asocia con clases asociación
 - 3.5. Elección de las directrices de las claves ajenas
4. Ejemplo
5. Teoría de la normalización

19

2.3. Clases especializadas



$A(a_0:t_{a_0}, \dots)$
 $CP: \{a_0\}$

$C(a_0:t_{a_0}, c_0:t_{c_0}, \dots)$
 $CP: \{a_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$

$B(a_0:t_{a_0}, b_0:t_{b_0}, \dots)$
 $CP: \{a_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$

$D(a_0:t_{a_0}, d_0:t_{d_0}, \dots)$
 $CP: \{a_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$

20

2.3. Clases especializadas

RI_{Total}:

Todo valor que aparece en el atributo a_0 de A debe aparecer en el atributo a_0 de B , de C o de D .

RI_{Disjunta}:

No puede haber un mismo valor en el atributo a_0 de B y en el a_0 de C ; ni en el a_0 de B y en el a_0 de D ; ni en el atributo a_0 de C y en el a_0 de D .

$A(a_0:t_{a_0}, \dots)$
 $CP: \{a_0\}$

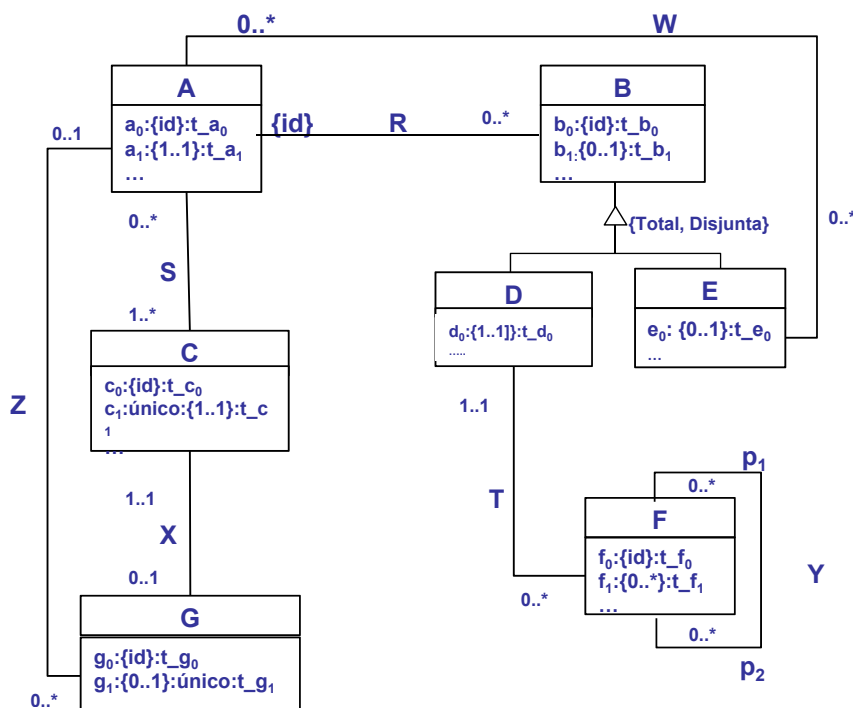
$C(a_0:t_{a_0}, c_0:t_{c_0}, \dots)$
 $CP: \{a_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$

$B(a_0:t_{a_0}, b_0:t_{b_0}, \dots)$
 $CP: \{a_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$

$D(a_0:t_{a_0}, d_0:t_{d_0}, \dots)$
 $CP: \{a_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$

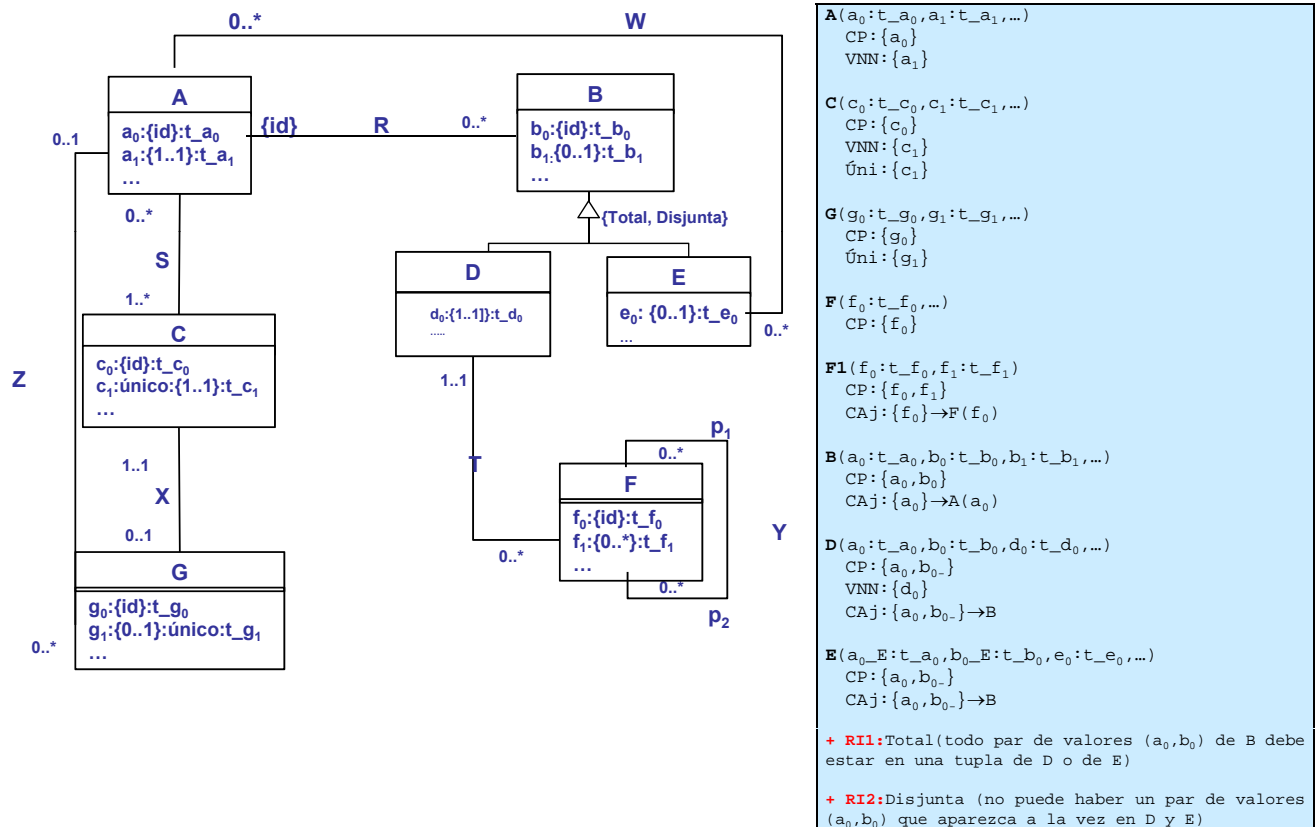
21

Ejemplo 1. Transformar sólo las clases



22

Ejemplo 1. Transformar sólo las clases



UD 4.3 Diseño lógico

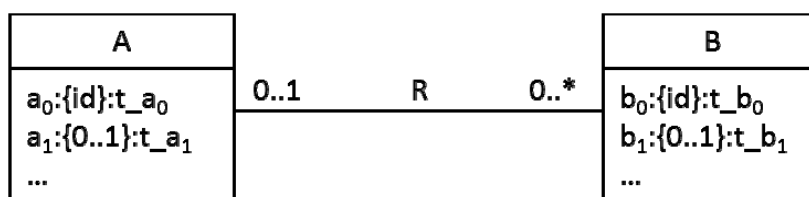
1. Introducción
2. Transformación de las clases
 - 2.1. Clases fuertes
 - 2.2. Clases débiles
 - 2.3. Clases especializadas
3. Transformación de las asociaciones
 - 3.1. No reflexivas
 - 3.2. Reflexivas
 - 3.3. Atributos de enlace
 - 3.4. Transformación de la asociación cuando se asocia con clases asociación
 - 3.5. Elección de las directrices de las claves ajenas
4. Ejemplo
5. Teoría de la normalización

UD 4.3 Diseño lógico

1. Introducción
2. Transformación de las clases
 - 2.1. Clases fuertes
 - 2.2. Clases débiles
 - 2.3. Clases especializadas
3. Transformación de las asociaciones
 - 3.1. No reflexivas
 - 3.2. Reflexivas
 - 3.3. Atributos de enlace
 - 3.4. Transformación de la asociación cuando se asocia con clases asociación
 - 3.5. Elección de las directrices de las claves ajenas
4. Ejemplo
5. Teoría de la normalización

25

3.1. Asociaciones binarias 0..1:0..*



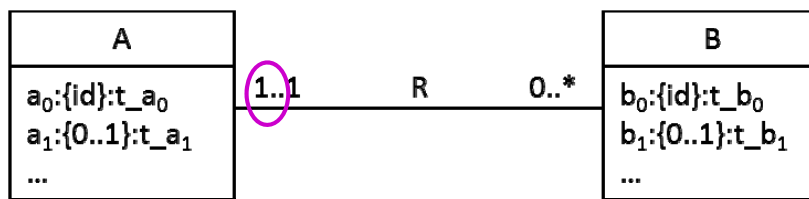
$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$
 $CP: \{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots, a_0:t_{a_0})$
 $CP: \{b_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$

SOLO SE REPRESENTA EN UN LADO, Y SE ESCOGE EL DE CARDINALIDAD MÍNIMA. En este caso, entre (x..1) y (x..*) se escoge representarlo en B usando (x..1)

26

3.1. Asociaciones binarias 1..1:0..*



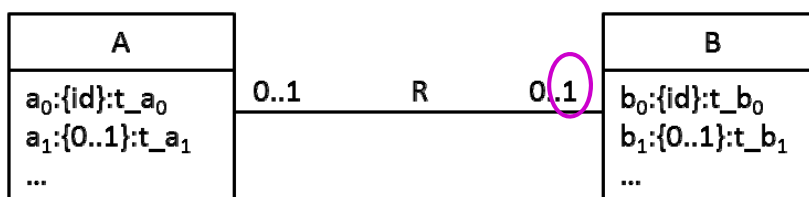
$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$
 $CP: \{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots, a_0:t_{a_0})$
 $CP: \{b_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$
 $VNN: \{a_0\}$

IGUAL QUE EL ANTERIOR añadiendo VNN.

27

3.1. Asociaciones binarias 0..1:0..1



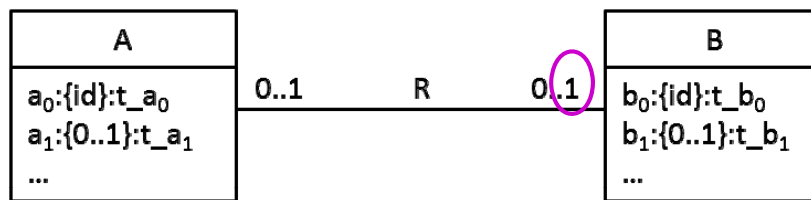
Esquema 1

$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$
 $CP: \{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots, a_0:t_{a_0})$
 $CP: \{b_0\}$
 $\text{Único}: \{a_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$

28

3.1. Asociaciones binarias 0..1:0..1



Esquema 2

$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots, \mathbf{b_0:t_{b_0}})$

CP: $\{a_0\}$

Único: $\{b_0\}$

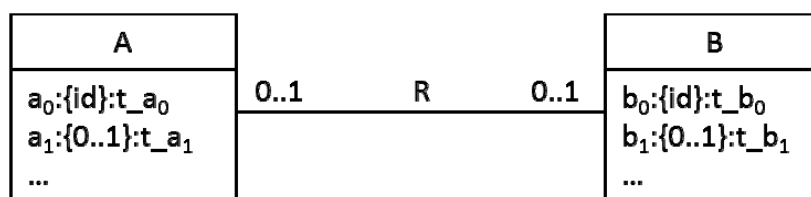
CAj: $\{b_0\} \rightarrow B(b_0)$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots)$

CP: $\{b_0\}$

29

3.1. Asociaciones binarias 0..1:0..1



Esquema 3

$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$

CP: $\{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots)$

CP: $\{b_0\}$

$R(\mathbf{b_0:t_{b_0}}, \mathbf{a_0:t_{a_0}})$

CP: $\{b_0\}$

Único: $\{a_0\}$

VNN: $\{a_0\}$

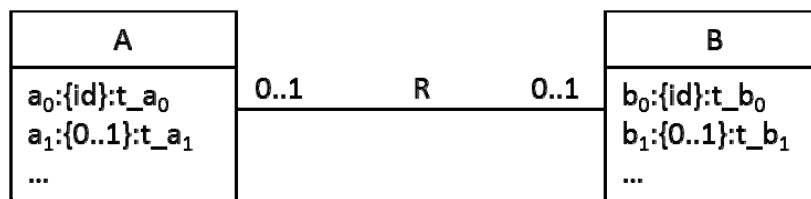
CAj: $\{a_0\} \rightarrow A(a_0)$

CAj: $\{b_0\} \rightarrow B(b_0)$

Hay más relaciones:
Es menos adecuado

30

3.1. Asociaciones binarias 0..1:0..1



Esquema 4

$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$
 $CP: \{a_0\}$

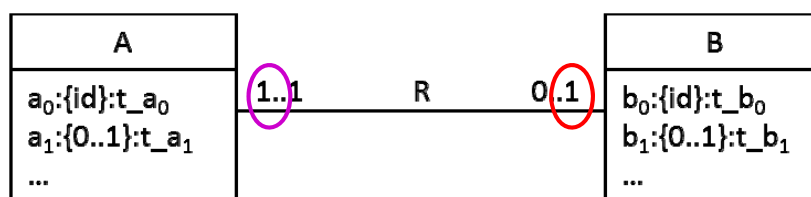
$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots)$
 $CP: \{b_0\}$

$R(b_0:t_{b_0}, a_0:t_{a_0})$
 $CP: \{a_0\}$
 $\text{Único}: \{b_0\}$
 $VNN: \{b_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$
 $CAj: \{b_0\} \rightarrow B(b_0)$

Hay más relaciones:
 Es menos adecuado

31

3.1. Asociaciones binarias 1..1:0..1

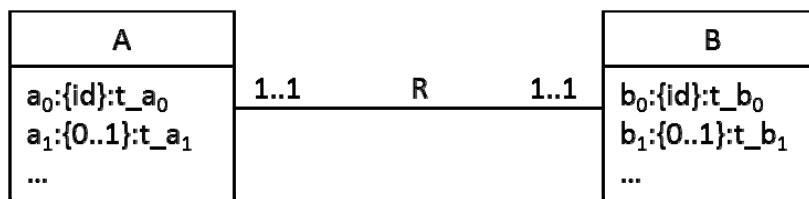


$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$
 $CP: \{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots, a_0:t_{a_0})$
 $CP: \{b_0\}$
 $\text{Único}: \{a_0\}$
 $VNN: \{a_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$

32

3.1. Asociaciones binarias 1..1:1..1



Esquema 1

$A-B(a_0:t_{a_0}, a_1:t_{a_1}, \dots, b_0:t_{b_0}, b_1:t_{b_1}, \dots)$

CP: {a₀}

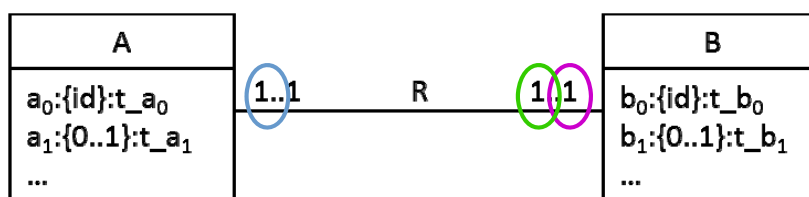
Único: {b₀}

VNN: {b₀}

Objetos de A-B más complicados de manipular

33

3.1. Asociaciones binarias 1..1:1..1



Esquema 2

$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$

CP: {a₀}

CAj: {a₀} → B(a₀)

Notar que la CAj es posible porque a₀ en B tiene restricción de valor único

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots, a_0:t_{a_0})$

CP: {b₀}

Único: {a₀}

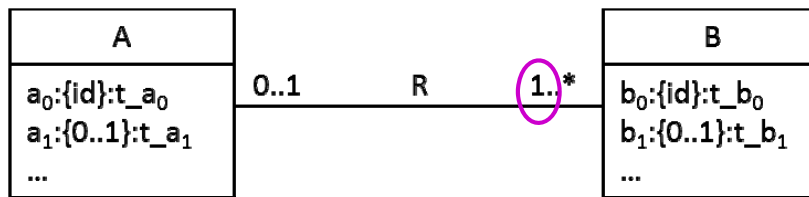
VNN: {a₀}

CAj: {a₀} → (a₀)

Mejor esquema

34

3.1. Asociaciones binarias 0..1:1..*



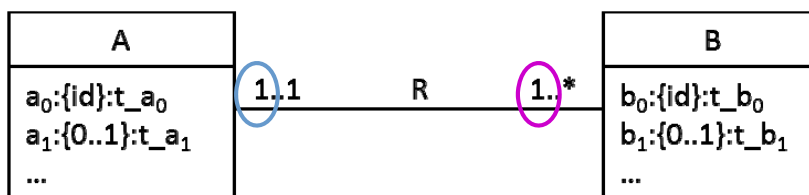
$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$
 CP: $\{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots, a_0:t_{a_0})$
 CP: $\{b_0\}$
 CAj: $\{a_0\} \rightarrow A(a_0)$

RI1: Todo valor que aparece en el atributo a_0 de A debe aparecer en el atributo a_0 de B.

35

3.1. Asociaciones binarias 1..1:1..*



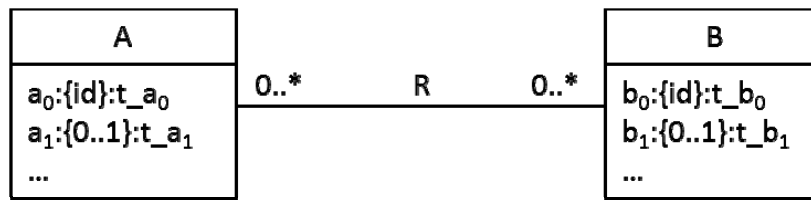
$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$
 CP: $\{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots, a_0:t_{a_0})$
 CP: $\{b_0\}$
 CAj: $\{a_0\} \rightarrow A(a_0)$
 VNN: $\{a_0\}$

RI1: Todo valor que aparece en el atributo a_0 de A debe aparecer en el atributo a_0 de B.

36

3.1. Asociaciones binarias 0..*:0..*



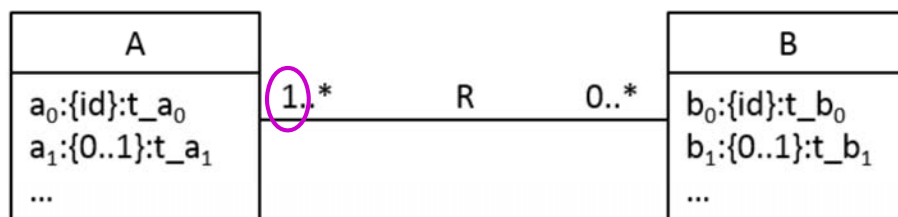
$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$
 $CP: \{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots)$
 $CP: \{b_0\}$

$R(a_0:t_{a_0}, b_0:t_{b_0})$
 $CP: \{a_0, b_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$
 $CAj: \{b_0\} \rightarrow B(b_0)$

37

3.1. Asociaciones binarias 1..*:0..*



$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$
 $CP: \{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots)$
 $CP: \{b_0\}$

$R(a_0:t_{a_0}, b_0:t_{b_0})$
 $CP: \{a_0, b_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$
 $CAj: \{b_0\} \rightarrow B(b_0)$

RI1: Todo valor que aparece en el atributo b_0 de B debe aparecer en el atributo b_0 de R .

38

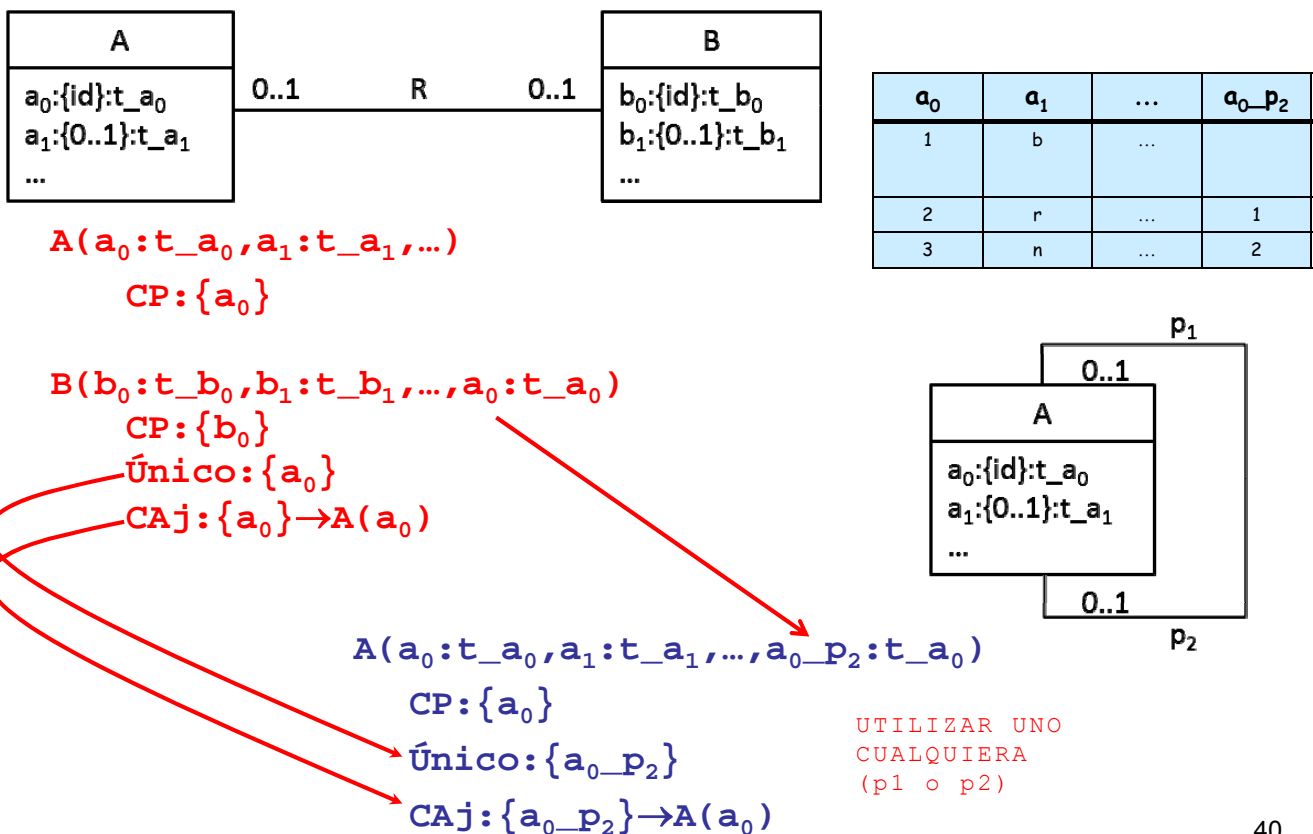
UD 4.3 Diseño lógico

1. Introducción
2. Transformación de las clases
 - 2.1. Clases fuertes
 - 2.2. Clases débiles
 - 2.3. Clases especializadas
3. Transformación de las asociaciones
 - 3.1. No reflexivas
 - 3.2. Reflexivas
 - 3.3. Atributos de enlace
 - 3.4. Transformación de la asociación cuando se asocia con clases asociación
 - 3.5. Elección de las directrices de las claves ajenas
4. Ejemplo
5. Teoría de la normalización

-> SOLO SE REPRESENTAN EN UN SENTIDO

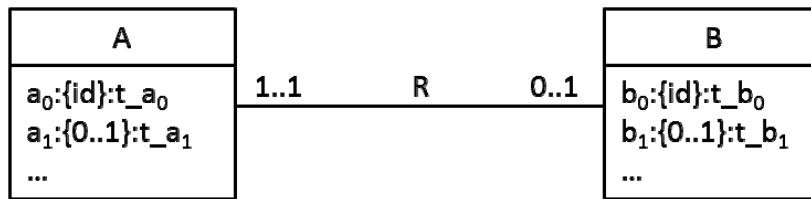
39

3.2. Asociaciones reflexivas 0..1:0..1



40

3.2. Asociaciones reflexivas 0..1:1..1



$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$
 $CP: \{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots, a_0:t_{a_0})$
 $CP: \{b_0\}$

$\text{Único}: \{a_0\}$

$VNN: \{a_0\}$

$CAj: \{a_0\} \rightarrow A(a_0)$

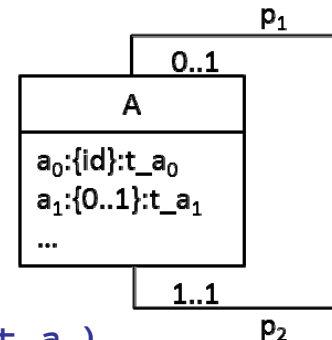
$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots, a_{0_p_2}:t_{a_0})$

$CP: \{a_0\}$

$\text{Único}: \{a_{0_p_2}\}$

$VNN: \{a_{0_p_2}\}$

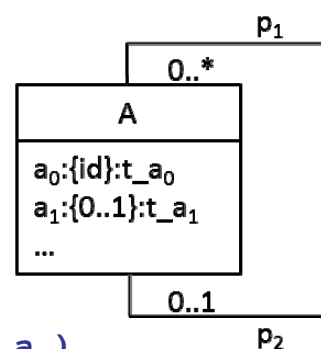
$CAj: \{a_{0_p_2}\} \rightarrow A(a_0)$



Utilizar el $\{1..1\}$

41

3.2. Asociaciones reflexivas 0..1:0..*



$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots, a_{0_p_2}:t_{a_0})$

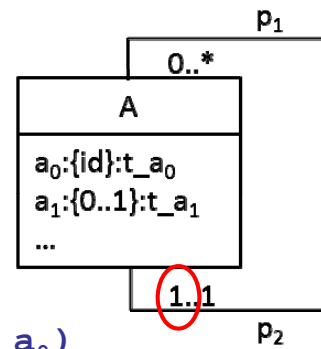
$CP: \{a_0\}$

$CAj: \{a_{0_p_2}\} \rightarrow A(a_0)$

UTILIZAR EL $\{0..1\}$

42

3.2. Asociaciones reflexivas 1..1:0..*



$A(a_0:t_a_0, a_1:t_a_1, \dots, a_{0_p_2}:t_a_0)$

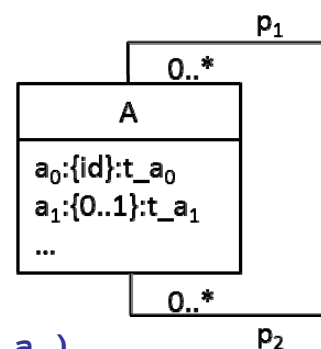
CP: $\{a_0\}$

CAj: $\{a_{0_p_2}\} \rightarrow A(a_0)$

VNN: $\{a_{0_p_2}\}$

43

3.2. Asociaciones reflexivas 0..*:0..*



$A(a_0:t_a_0, a_1:t_a_1, \dots, a_{0_p_2}:t_a_0)$

CP: $\{a_0\}$

$R(a_{0_p_1}:t_a_0, a_{0_p_2}:t_a_0)$

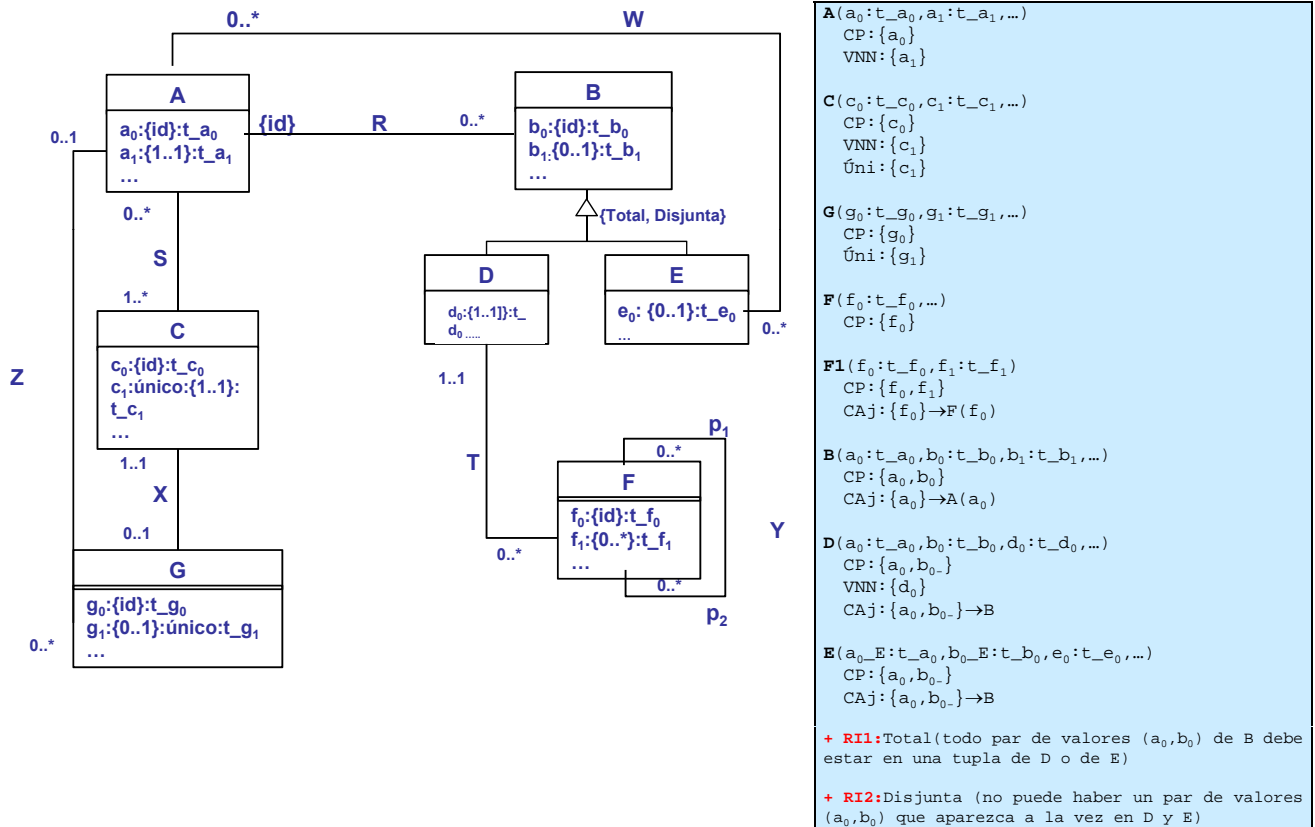
CP: $\{a_{0_p_1}, a_{0_p_2}\}$

CAj: $\{a_{0_p_1}\} \rightarrow A(a_0)$

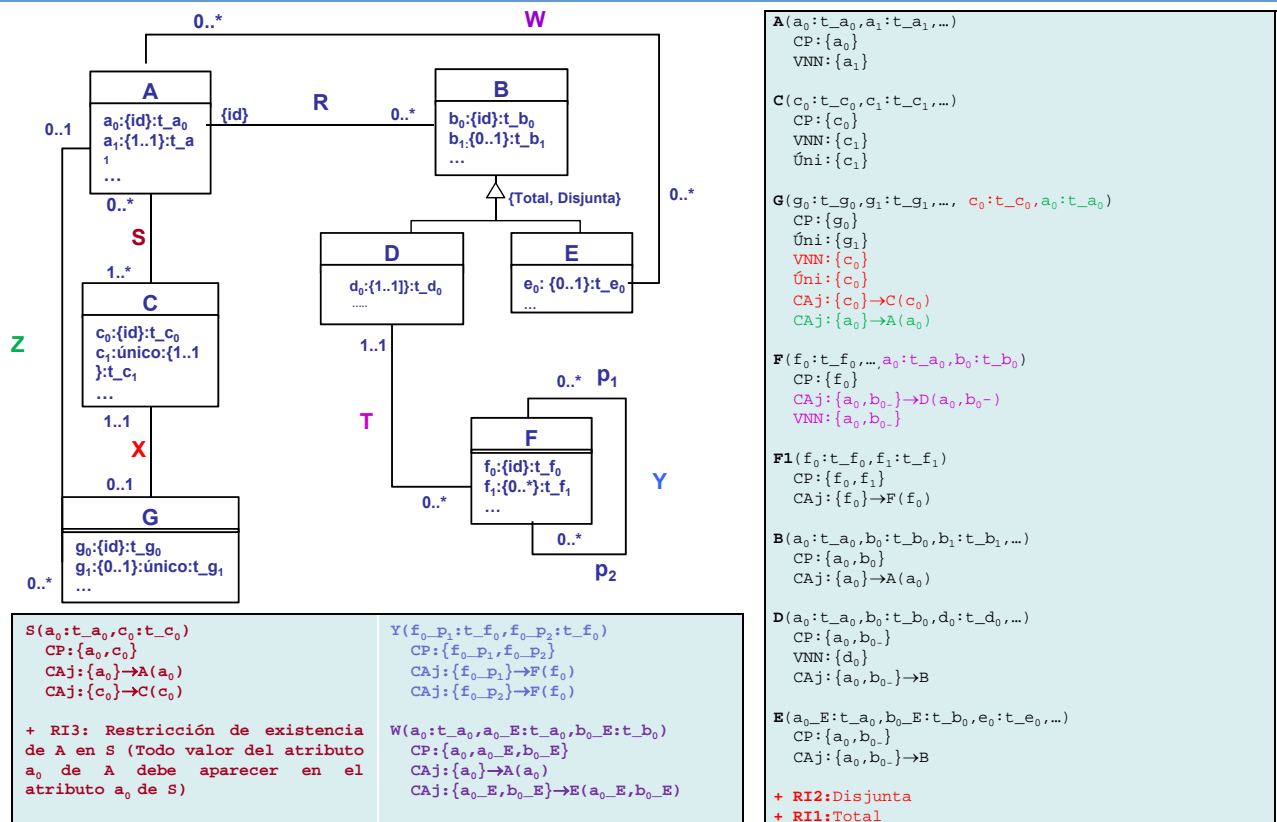
CAj: $\{a_{0_p_2}\} \rightarrow A(a_0)$

44

Ejemplo 1. Transformar las asociaciones del Diagrama



Ejemplo 1. Transformar las asociaciones del Diagrama



UD 4.3 Diseño lógico

1. Introducción
2. Transformación de las clases
 - 2.1. Clases fuertes
 - 2.2. Clases débiles
 - 2.3. Clases especializadas
3. Transformación de las asociaciones
 - 3.1. No reflexivas
 - 3.2. Reflexivas
 - 3.3. Atributos de enlace
 - 3.4. Transformación de la asociación cuando se asocia con clases asociación
 - 3.5. Elección de las directrices de las claves ajenas
4. Ejemplo
5. Teoría de la normalización

48

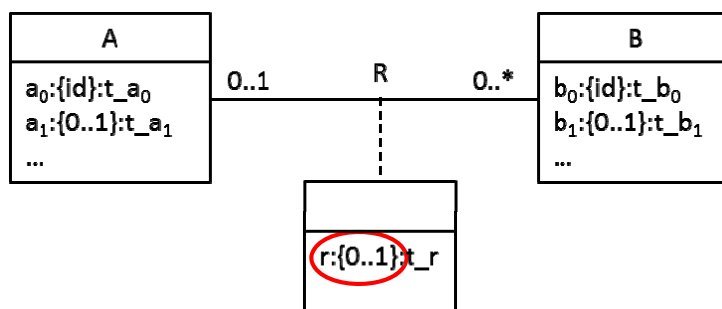
3.3. Asociación con atributos de enlace

- Los atributos de enlace se incluyen en la tabla donde está representada la asociación que describen.
- La presencia de atributos de enlace puede hacer que el esquema aplicado (según secciones anteriores) deje de ser adecuado:
 - Añadir Restricciones de Integridad (R.I.)
 - Añadir nueva tabla

49

3.3. Asociación con atributos de enlace

b)



$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$

CP: $\{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots, a_0:t_{a_0}, r:t_r)$

CP: $\{b_0\}$

CAj: $\{a_0\} \rightarrow A(a_0)$

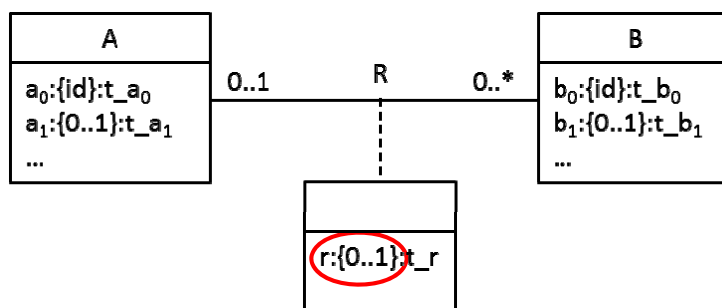
MEJOR EL DE TRANSPARENCIA
SIGUIENTE

RI1: No puede existir una tupla en B que tenga el atributo a_0 nulo y el atributo r distinto de nulo

50

3.3. Asociación con atributos de enlace

b)



$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$

CP: $\{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots)$

CP: $\{b_0\}$

$R(b_0:t_{b_0}, a_0:t_{a_0}, r:t_r)$

CP: $\{b_0\}$

VNN: $\{a_0\}$

CAj: $\{a_0\} \rightarrow A(a_0)$

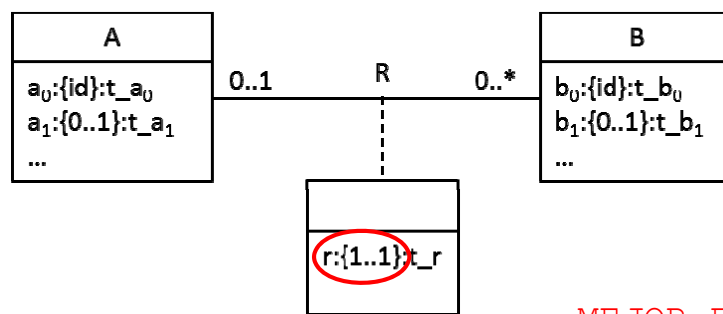
CAj: $\{b_0\} \rightarrow B(b_0)$

Mejor solución:
No incluye R.I.

51

3.3. Asociación con atributos de enlace

a)



$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$

CP: $\{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots, a_0:t_{a_0}, r:t_r)$

CP: $\{b_0\}$

CAj: $\{a_0\} \rightarrow A(a_0)$

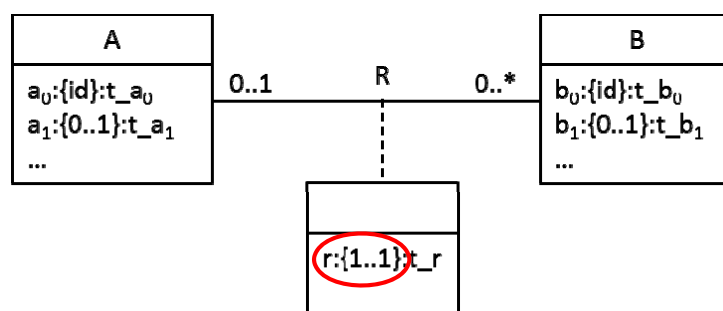
MEJOR EL DE TRANSPARENCIA
SIGUIENTE

RI1: En toda tupla de B se debe cumplir
que o bien a_0 y r son no nulo o a_0 y r son
nulos.

52

3.3. Asociación con atributos de enlace

a)



$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$

CP: $\{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots)$

CP: $\{b_0\}$

$R(b_0:t_{b_0}, a_0:t_{a_0}, r:t_r)$

CP: $\{b_0\}$

VNN: $\{a_0\}$

VNN: $\{r\}$

CAj: $\{a_0\} \rightarrow A(a_0)$

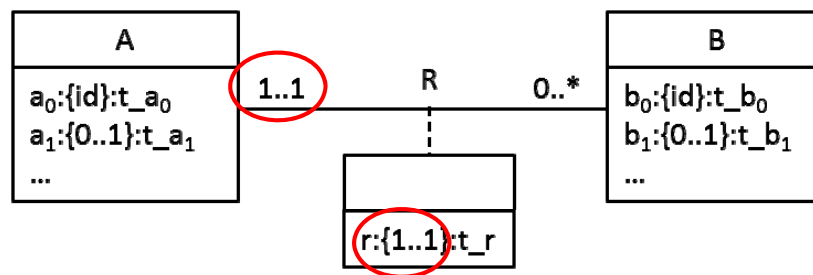
CAj: $\{b_0\} \rightarrow B(b_0)$

Mejor solución:
No incluye R.I.

53

3.3. Asociación con atributos de enlace

c)



$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$

CP: {a₀}

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots, a_0:t_{a_0}, \mathbf{r}:t_{\mathbf{r}})$

CP: {b₀}

VNN: {a₀}

VNN: {r}

CAj: {a₀} → A

54

UD 4.3 Diseño lógico

1. Introducción

2. Transformación de las clases

2.1. Clases fuertes

2.2. Clases débiles

2.3. Clases especializadas

3. Transformación de las asociaciones

3.1. No reflexivas

3.2. Reflexivas

3.3. Atributos de enlace

3.4. Transformación de la asociación cuando se asocia con clases asociación

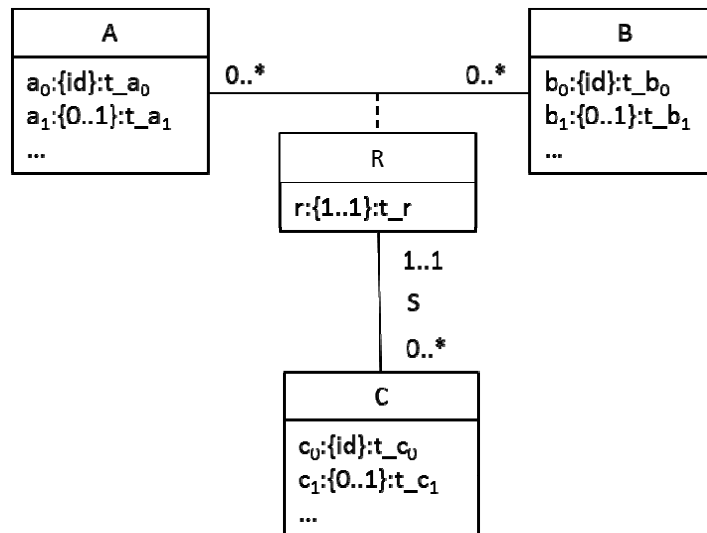
3.5. Elección de las directrices de las claves ajenas

4. Ejemplo

5. Teoría de la normalización

55

3.4. Asociación con clase asociación

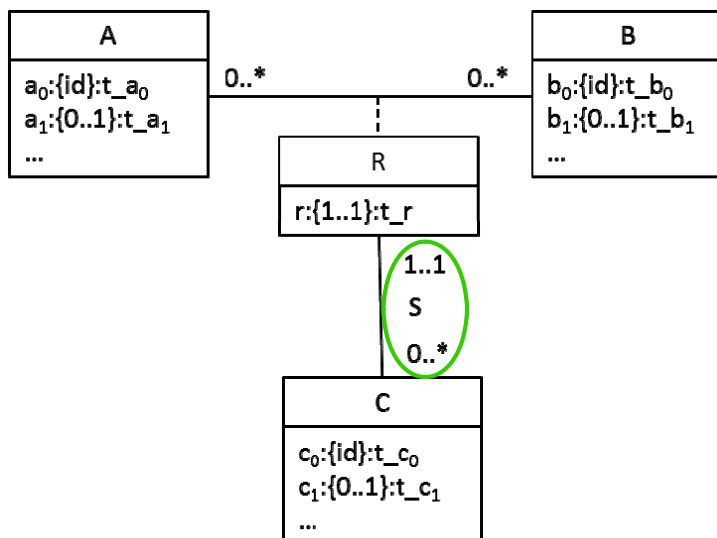


1. Comenzar el diseño transformando la asociación entre A y B (según esquemas anteriores)
2. Transformar el resto del diagrama (asociación S)
3. Comprobar validez

56

3.4. Asociación con clase asociación.

Ej1



$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$

CP: $\{a_0\}$

$B(b_0:t_{b_0}, b_1:t_{b_1}, \dots)$

CP: $\{b_0\}$

$R(a_0:t_{a_0}, b_0:t_{b_0}, \textcolor{red}{r}:t_r)$

CP: $\{a_0, b_0\}$

CAj: $\{a_0\} \rightarrow A(a_0)$

CAj: $\{b_0\} \rightarrow B(b_0)$

$\textcolor{red}{VNN: \{r\}}$

$C(c_0:t_{c_0}, c_1:t_{c_1}, \dots, \textcolor{green}{a_0:t_{a_0}}, \textcolor{green}{b_0:t_{b_0}})$

CP: $\{c_0\}$

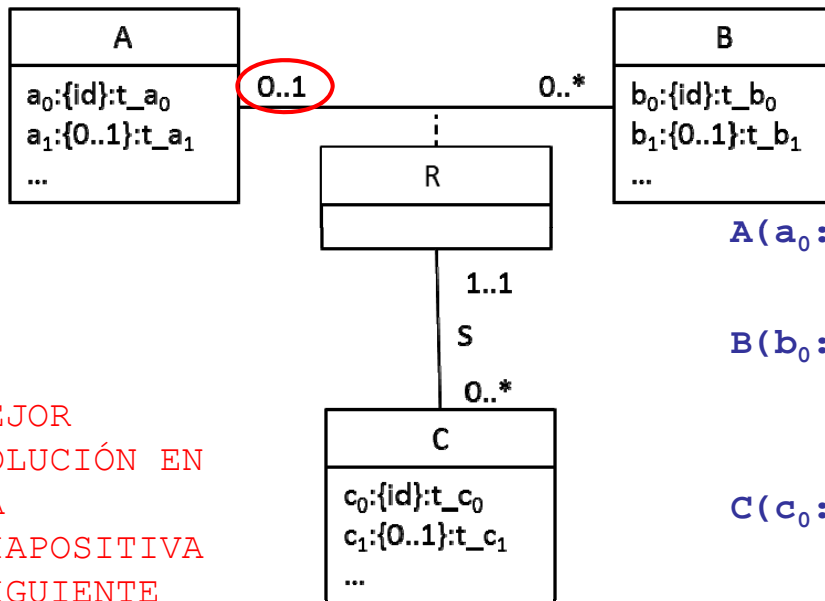
CAj: $\{a_0, b_0\} \rightarrow R(a_0, b_0)$

$\textcolor{green}{VNN: \{a_0, b_0\}}$

57

3.4. Asociación con clase asociación.

Ej2



Se puede mejorar

MEJOR
SOLUCIÓN EN
LA
DIAPOSITIVA
SIGUIENTE

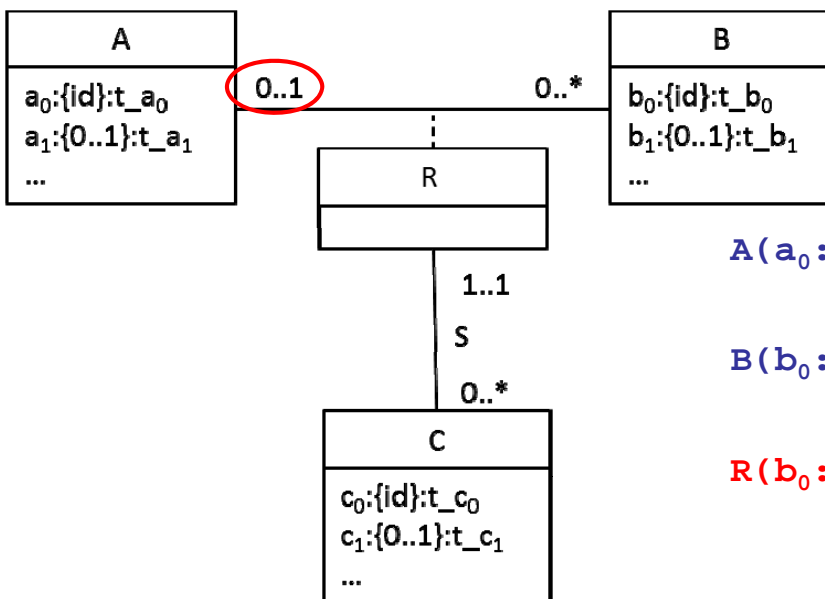
$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$
 $CP: \{a_0\}$
 $B(b_0:t_{b_0}, b_1:t_{b_1}, \dots, a_0:t_{a_0})$
 $CP: \{b_0\}$
 $CAj: \{a_0\} \rightarrow A$
 $C(c_0:t_{c_0}, c_1:t_{c_1}, \dots, b_0:t_{b_0})$
 $CP: \{c_0\}$
 $CAj: \{b_0\} \rightarrow B$
 $VNN: \{b_0\}$

RI1: No puede existir una tupla en C que se relacione con una tupla de B que no este relacionada con A .

58

3.4. Asociación con clase asociación.

Ej2



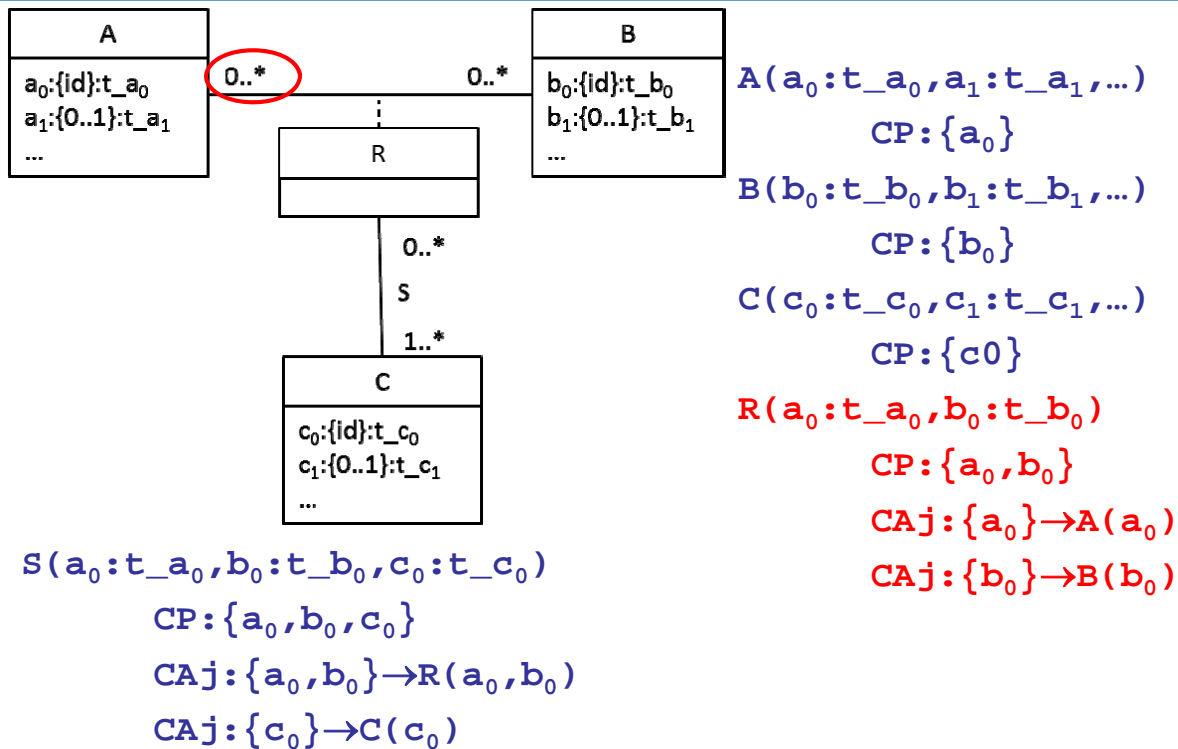
Mejor solución:
No tiene R.I.

$A(a_0:t_{a_0}, a_1:t_{a_1}, \dots)$
 $CP: \{a_0\}$
 $B(b_0:t_{b_0}, b_1:t_{b_1}, \dots)$
 $CP: \{b_0\}$
 $R(b_0:t_{b_0}, a_0:t_{a_0})$
 $CP: \{b_0\}$
 $VNN: \{a_0\}$
 $CAj: \{a_0\} \rightarrow A(a_0)$
 $CAj: \{b_0\} \rightarrow B(b_0)$
 $C(c_0:t_{c_0}, c_1:t_{c_1}, \dots, b_0:t_{b_0})$
 $CP: \{c_0\}$
 $CAj: \{b_0\} \rightarrow R$
 $VNN: \{b_0\}$

59

3.4. Asociación con clase asociación.

Ej3

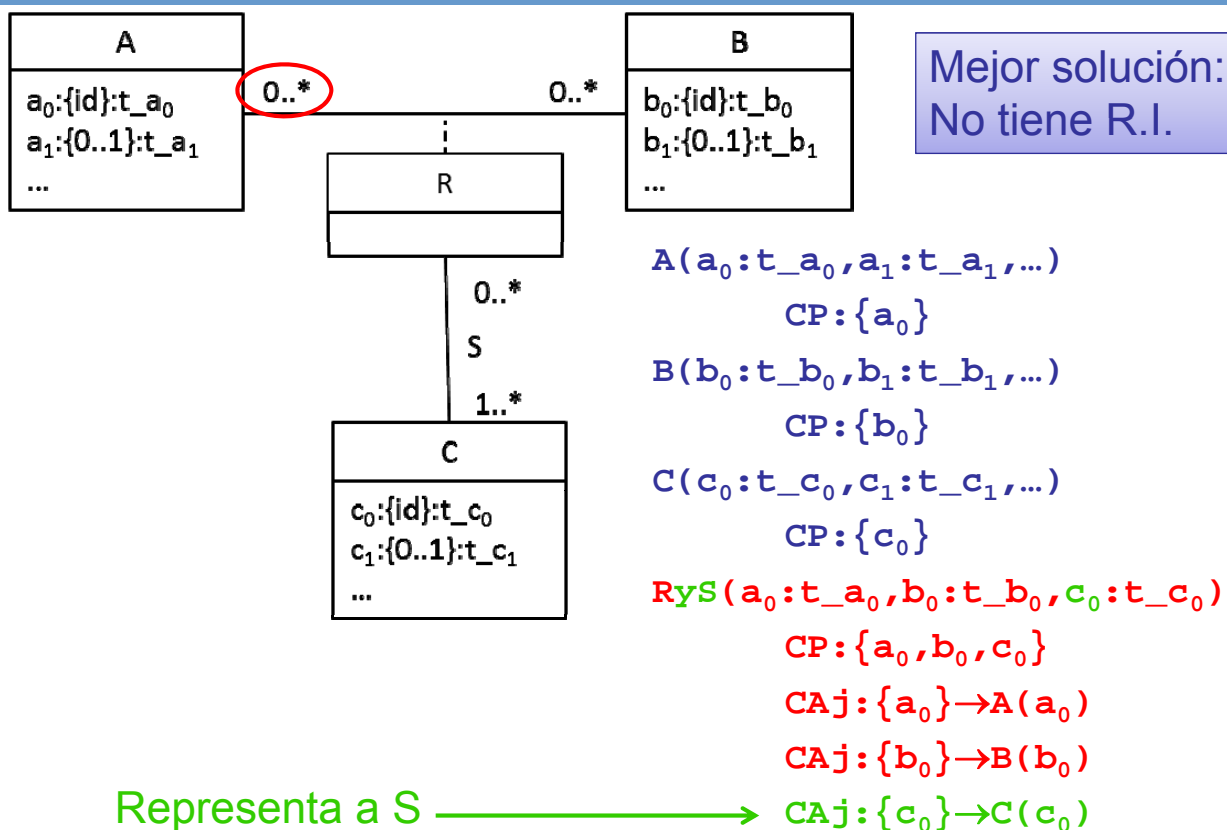


RI1: No puede existir una tupla en R tal que el valor de (a_0, b_0) , no aparezca en los atributos (a_0, b_0) de S.

60

3.4. Asociación con clase asociación.

Ej3



61

UD 4.3 Diseño lógico

1. Introducción
2. Transformación de las clases
 - 2.1. Clases fuertes
 - 2.2. Clases débiles
 - 2.3. Clases especializadas
3. Transformación de las asociaciones
 - 3.1. No reflexivas
 - 3.2. Reflexivas
 - 3.3. Atributos de enlace
 - 3.4. Transformación de la asociación cuando se asocia con clases asociación
 - 3.5. Elección de las directrices de las claves ajenas
4. Ejemplo
5. Teoría de la normalización

62

3.5 Elección de directrices CAj

- **Directrices de borrado y de actualización** para que el SGBD pueda, restaurar la integridad referencial cuando ésta se viola:
 - Restrictiva
 - A nulos
 - En cascada
- Estas directrices permiten la representación de algunas restricciones de integridad incluidas en el esquema conceptual o algunas detectadas en el análisis del sistema.
- A veces hay varias opciones razonables.

63

3.5 Elección de directrices CAj

A veces hay varias opciones razonables.

Directriz de actualización:

- En general se opta por la directriz en CASCADA

Directriz de borrado

- Si el valor referenciado tiene restricción VNN, “a nulos” no tiene sentido
- Atributos multivaluados (0..*) estarán en tabla referenciada: Mejor borrado “en cascada”

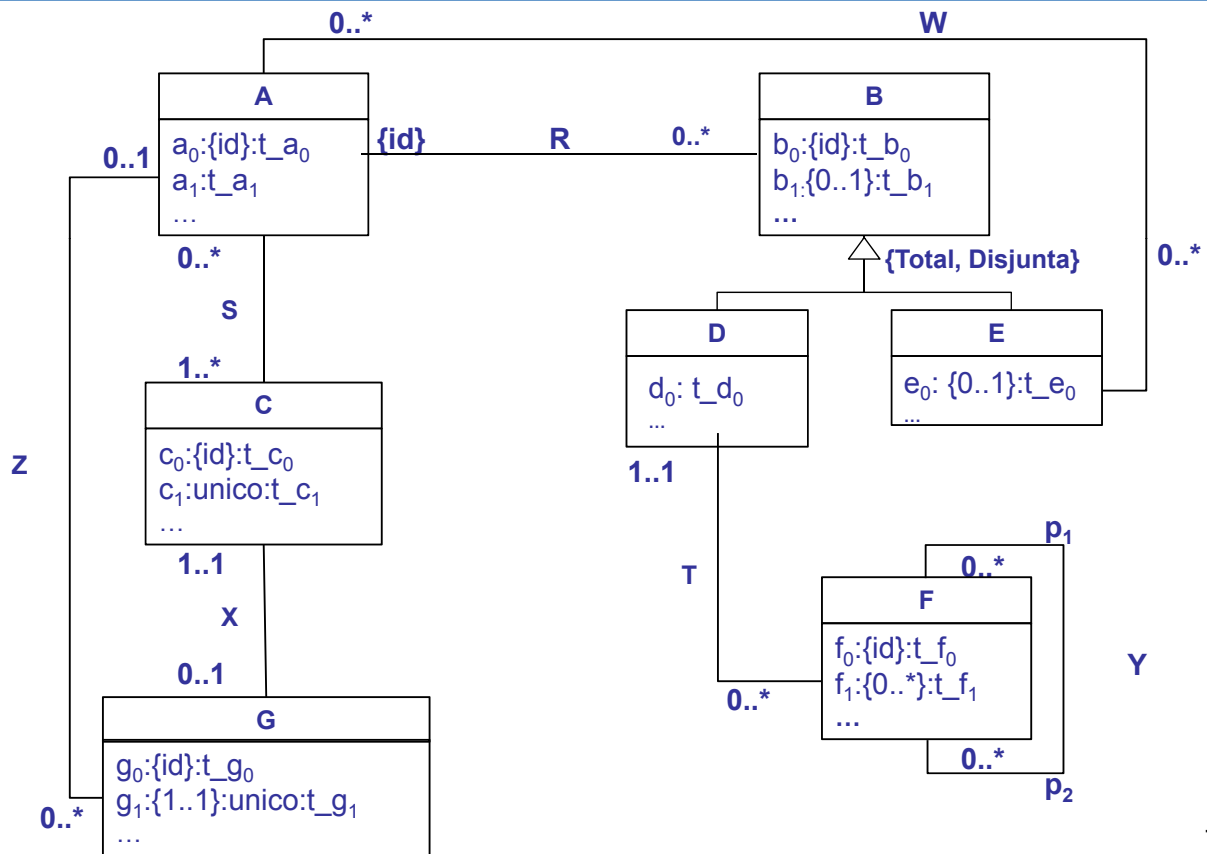
64

UD 4.3 Diseño lógico

1. Introducción
2. Transformación de las clases
 - 2.1. Clases fuertes
 - 2.2. Clases débiles
 - 2.3. Clases especializadas
3. Transformación de las asociaciones
 - 3.1. No reflexivas
 - 3.2. Reflexivas
 - 3.3. Atributos de enlace
 - 3.4. Transformación de la asociación cuando se asocia con clases asociación
 - 3.5. Elección de las directrices de las claves ajenas
4. Ejemplo
5. Teoría de la normalización

74

4 Ejemplo



75

4 Ejemplo. Transformación de clases

$A(a_0: t_{a_0}, a_1: t_{a_1}, \dots)$

CP: $\{a_0\}$

$B(b_0: t_{b_0}, b_1: t_{b_1}, \dots, a_0: t_{a_0})$

CP: $\{a_0, b_0\}$

CAj: $\{a_0\} \rightarrow A(a_0)$

B débil

$C(c_0: t_{c_0}, c_1: t_{c_1}, \dots)$

CP: $\{c_0\}$

UNI: $\{c_1\}$

$D(a_0: t_{a_0}, b_0: t_{b_0}, d_0: t_{d_0}, \dots)$

CP: $\{a_0, b_0\}$

CAj: $\{a_0, b_0\} \rightarrow B(a_0, b_0)$

D especializa B

$E(a_0: t_{a_0}, b_0: t_{b_0}, e_0: t_{e_0}, \dots)$

CP: $\{a_0, b_0\}$

CAj: $\{a_0, b_0\} \rightarrow B(a_0, b_0)$

E especializa B

IC_{Total}: "Todo par a_0, b_0 que esté en B, debe estar en D o E".

IC_{Disjunta}: "Un par a_0, b_0 no puede estar en más de una tupla de D o E".

$F(f_0: t_{f_0}, \dots)$

CP: $\{f_0\}$

f_1 es multivaluado

$F_{f1}(f_0: t_{f_0}, f_1: t_{f_1})$

CP: $\{f_0, f_1\}$

CAj: $\{f_0\} \rightarrow F(f_0)$

$G(g_0: t_{g_0}, g_1: t_{g_1}, \dots)$

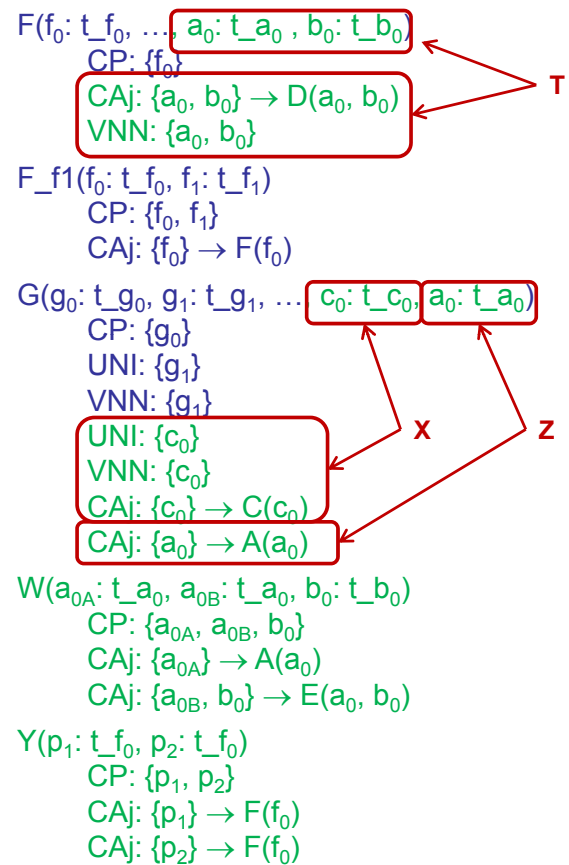
CP: $\{g_0\}$

UNI: $\{g_1\}$

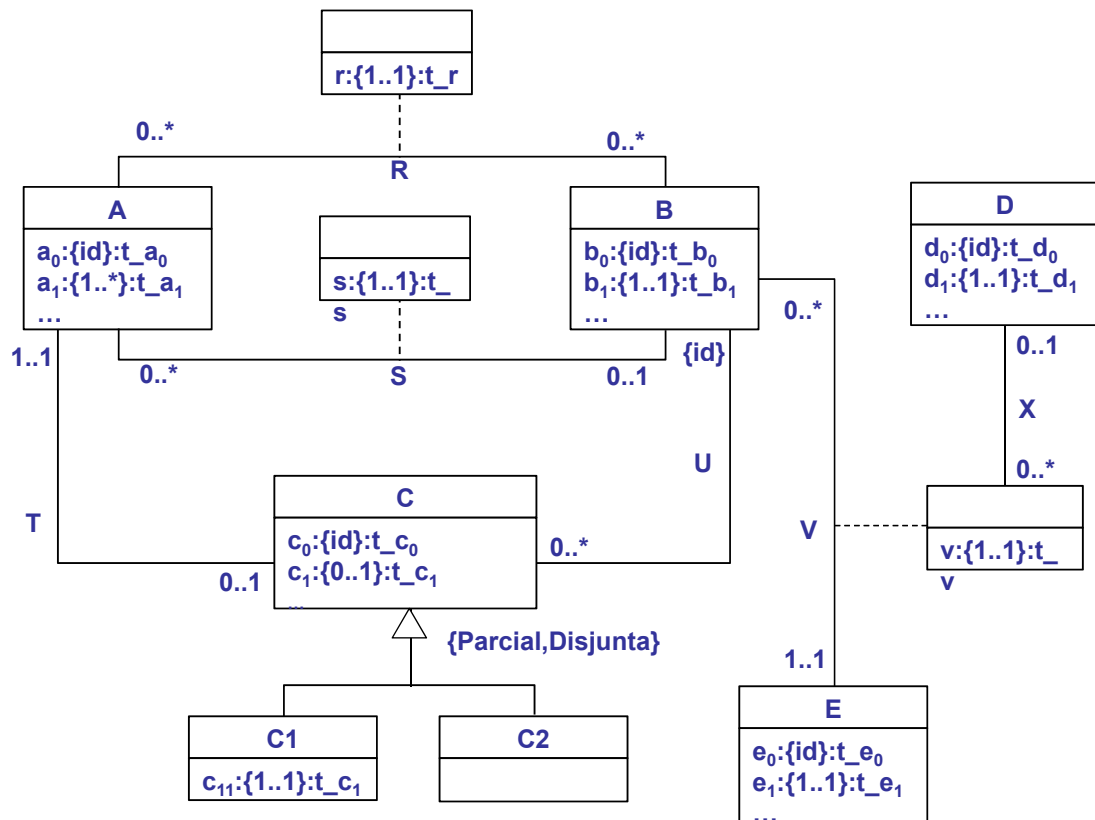
VNN: $\{g_1\}$

76

$A(a_0: t_{a_0}, a_1: t_{a_1}, \dots)$
 CP: $\{a_0\}$
 $B(b_0: t_{b_0}, b_1: t_{b_1}, \dots, a_0: t_{a_0})$
 CAj: $\{a_0, b_0\}$
 CAj: $\{a_0\} \rightarrow A(a_0)$
 $C(c_0: t_{c_0}, c_1: t_{c_1}, \dots)$
 CP: $\{c_0\}$
 UNI: $\{c_1\}$
 $D(a_0: t_{a_0}, b_0: t_{b_0}, d_0: t_{d_0}, \dots)$
 CP: $\{a_0, b_0\}$
 CAj: $\{a_0, b_0\} \rightarrow B(a_0, b_0)$
 $E(a_0: t_{a_0}, b_0: t_{b_0}, e_0: t_{e_0}, \dots)$
 CP: $\{a_0, b_0\}$
 CAj: $\{a_0, b_0\} \rightarrow B(a_0, b_0)$
 IC_{Total} : "Todo par a_0, b_0 que esté en B , debe estar en D o E ".
 $IC_{Disjunta}$: "Un par a_0, b_0 no puede estar en más de una tupla de D o E "
 $S(a_0: t_{a_0}, c_0: t_{c_0})$
 CP: $\{a_0, c_0\}$
 CAj: $\{a_0\} \rightarrow A(a_0)$
 CAj: $\{c_0\} \rightarrow C(c_0)$
 IC : "Todo a_0 de A debe aparecer al menos una vez en S "



4. Ejemplo Parcial 2 2012



UD 3.3 Diseño lógico

1. Introducción
2. Transformación de las clases
 - 2.1. Clases fuertes
 - 2.2. Clases débiles
 - 2.3. Clases especializadas
3. Transformación de las asociaciones
 - 3.1. No reflexivas
 - 3.2. Reflexivas
 - 3.3. Atributos de enlace
 - 3.4. Transformación de la asociación cuando se asocia con clases asociación
 - 3.5. Elección de las directrices de las claves ajenas
4. Ejemplo
5. Teoría de la normalización

83

5. Teoría de la Normalización

Antes de considerarlo definitivo, el esquema lógico obtenido con las transformaciones presentadas debe ser revisado para comprobar que se encuentra adecuadamente diseñado.



Normalizar el esquema lógico aplicando la
Teoría de la Normalización.



Proceso durante el cual los esquemas de relaciones insatisfactorios se descomponen repartiendo sus atributos entre esquemas de relación más pequeños que poseen propiedades deseables.

84

5. Teoría de la Normalización

- **Conceptos Previos:**

- Dependencia Funcional (completa o no)
- Diagrama de Dependencias Funcionales
- Clave de una relación
- Atributo Primo

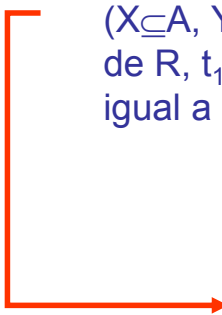
85

5. Teoría de la Normalización

Dependencia Funcional (completa o no)

Sea $A = \{A_1, \dots, A_n\}$ conjunto de atributos del esquema R

Una **dependencia funcional** entre dos conjuntos de atributos X e Y ($X \subseteq A, Y \subseteq A, X \neq Y$), $X \rightarrow Y \Rightarrow$ para cualquier par de tuplas posibles de R, t_1 y t_2 , se cumple que si $t_1[X]$ es igual a $t_2[X]$, entonces $t_1[Y]$ es igual a $t_2[Y]$.



Para un valor de X, Y sólo puede tomar un valor posible

86

5. Teoría de la Normalización

Dependencia Funcional **completa**

Una dependencia funcional entre dos conjuntos de atributos $X \rightarrow Y$ es completa si la eliminación de cualquier atributo A_i de X hace que la dependencia deje de existir, es decir si $\forall A_i / A_i \in X$ se cumple que Y no depende funcionalmente de $(X - \{A_i\})$.

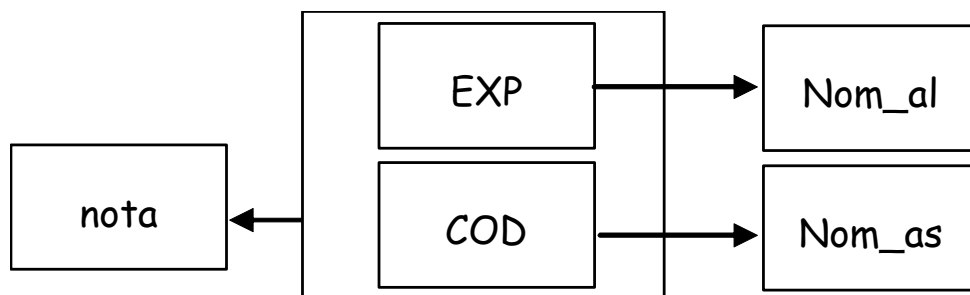
87

5. Teoría de la Normalización

Diagrama de Dependencias Funcionales

Representación gráfica de las dependencias. Utilizan cajas para enmarcar los atributos o conjuntos de atributos y flechas para denotar la dependencia funcional.

Normalmente sólo se representan las dependencias funcionales completas.



88

5. Teoría de la Normalización

Clave de una relación

Sea R un esquema de relación cuyo conjunto de atributos es $A = \{A_1, A_2, \dots, A_n\}$, y sea C un subconjunto de atributos de ese esquema ($C \subseteq A$); se dice que C es una **clave** de R si C es la clave primaria de R o bien si C tiene una restricción de unicidad.



Todos los atributos de una relación dependen funcionalmente de cada clave que hay en la relación

89

5. Teoría de la Normalización

Atributo Primo

Sea R un esquema de relación cuyo conjunto de atributos es $A = \{A_1, A_2, \dots, A_n\}$, un atributo A_i se dice que **es primo** si forma parte de alguna clave de R

90

1ª Forma normal

Una Relación está en 1FN si **sus atributos solamente toman valores atómicos** (simples e indivisibles).

- Problemas de utilizar relaciones que no están en 1FN: Hay que utilizar operadores asociados a los tipos de datos complejos (listas, conjuntos, registros,...)

CP: {vcod}

		Conjunto	Registro
vcod	nombre	teléfonos	dir
V1	Pepe	(96 3233258, 964 523844, 979 568987, 987 456123)	Paz 7, Valencia
V2	Juan	(96 3852741, 910 147258)	Eolo 3, Castellón
V3	Eva	(987 456 312)	F. Lorca 2, Utiel

91

1ª Forma normal

➤ Paso a 1FN:

- si **R** tiene un atributo que es un conjunto:



- eliminarlo de la relación y definir una nueva relación con el atributo no-atómico y los atributos que contiene la clave primaria de R, luego buscar la clave primaria de la nueva relación.

92

1ª Forma normal

vcod	nombre	teléfonos	dir
V1	Pepe	(96 3233258, 964 523844, 979 568987, 987 456123)	Paz 7, Valencia
V2	Juan	(96 3852741, 910 147258)	Eolo 3, Castellón
V3	Eva	(987 456 312)	F. Lorca 2, Utiel

vcod	nombre	dir
V1	Pepe	Paz 7, Valencia
V2	Juan	Eolo 3, Castellón
V3	Eva	F. Lorca 2, Utiel

vcod	teléfonos
V1	96 3233258
V2	96 3852741
V3	987 456 312
V1	964 523844
V1	979 568987
V1	987 456123
V2	910 147258

CP?

93

1ª Forma normal

Proveedor(vcod, nombre, teléfonos, dir)

CP: {vcod}

Proveedor(vcod, nombre, dir)

CP: {vcod}

Listín(vcod, teléfono)

CP: {teléfono}vcod

CAj: {vcod} → Proveedor

VNN: {vcod}

Si los teléfonos no se pueden compartir
Si los teléfonos se pueden compartir

94

1ª Forma normal

➤ Paso a 1FN:

- si R tiene un atributo que es un registro sustituirlo por los campos del registro.

vcod	nombre	dir
V1	Pepe	Paz 7, Valencia
V2	Juan	Eolo 3, Castellón
V3	Eva	F. Lorca 2, Utiel

A blue arrow points from the 'dir' column of the first table to the 'calle' column of the second table. An orange bracket groups the 'calle', 'número', and 'ciudad' columns of the second table, with a line connecting it to the 'dir' column of the first table.

vcod	nombre	calle	número	ciudad
V1	Pepe	Paz	7	Valencia
V2	Juan	Eolo	3	Castellón
V3	Eva	F. Lorca	2	Utiel

95

1ª Forma normal

Proveedor(vcod, nombre, teléfonos, dir)

CP: {vcod}

Proveedor(vcod, nombre, calle, número, ciudad)

CP: {vcod}

Listín(vcod, teléfono)

CP: {teléfono, vcod}

CAj: {vcod} → Proveedor

96

2ª Forma normal

Una Relación R está en 2FN si está en 1FN y todo atributo no-primo depende funcionalmente de forma **completa** de la clave primaria de R.

Si la CP está formada por un único atributo, significa que ya está en Segunda forma normal (y, por tanto, también en 1ª forma normal).

⌘ Problemas de utilizar relaciones que no están en 2FN

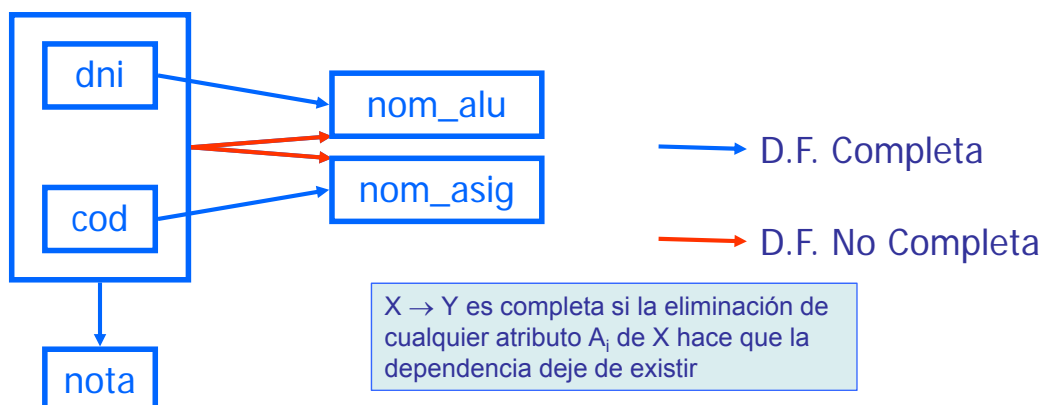
- Existen redundancias, con lo que se complica la manipulación de la información.
- No es fácil insertar ni borrar

97

2ª Forma normal

CP: {dni,cod}

dni	nom_alu	cod	nom_asig	nota
1	Pepe	DBD	Diseño de Bases de Datos	6
1	Pepe	BDA	Bases de Datos	7
2	Juana	DBD	Diseño de Bases de Datos	7
2	Juana	BDA	Bases de Datos	5



98

2ª Forma normal

➤ Paso a 2FN:

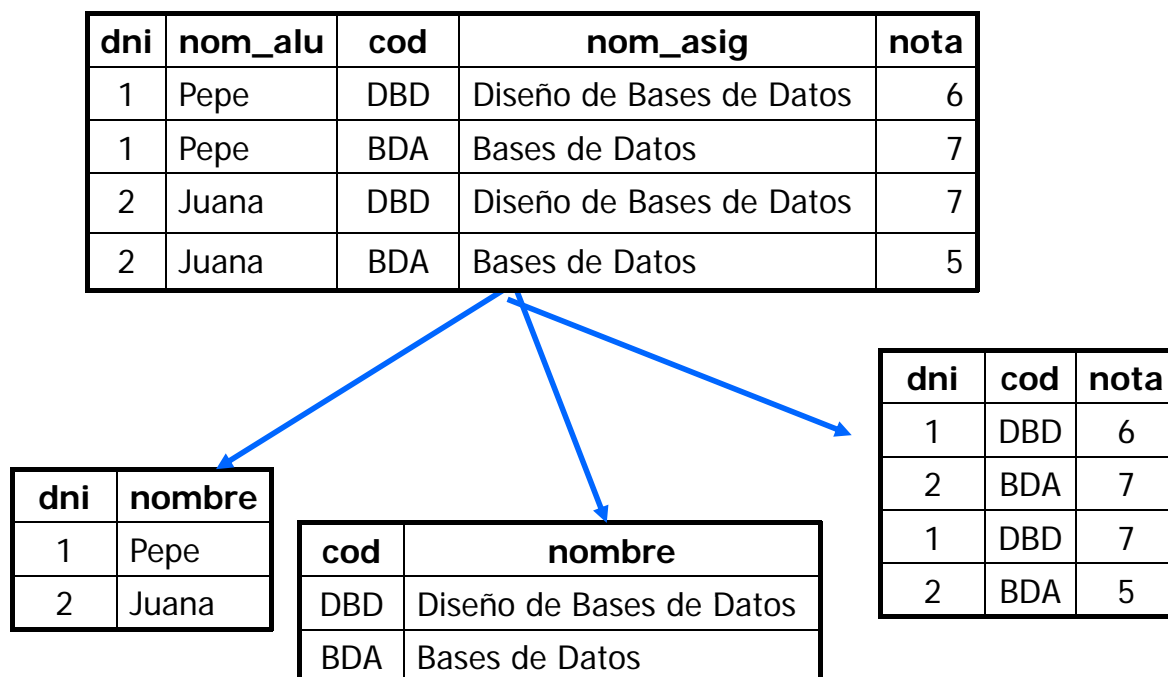
- La clave consta de más de un atributo y existe algún atributo no-primo que no depende completamente de la clave principal:



Dividir la relación original en relaciones para eliminar esas dependencias no completas.

99

2ª Forma normal



100

2ª Forma normal

Examina(dni, cod, nom_alu, nom_asig, nota)

CP: {dni,cod}

→ Alumno(dni, nom_alu)

CP: {dni}

→ Asignatura(cod, nom_asig)

CP: {cod}

→ Examina(dni, cod, nota)

CP: {dni,cod}

CAj: {dni} → Alumno

CAj: {cod} → Asignatura

101

3ª Forma normal

Una Relación está en 3FN si está en 2FN y **no hay dependencias funcionales entre atributos no-primos.**

- Problemas de utilizar relaciones que no están en 3FN
 - Existen redundancias, con lo que se complica la manipulación de la información.
 - No es fácil insertar ni borrar

Dependencia funcional transitiva

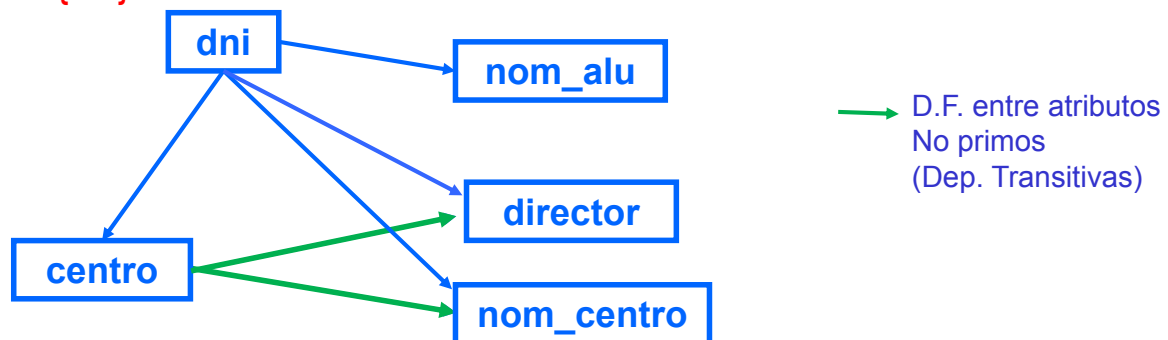
Sea R un esquema de relación cuyo conjunto de atributos es $A = \{A_1, A_2, \dots, A_n\}$, Si $\{A_i\} \rightarrow \{A_j\}$ y $\{A_j\} \rightarrow \{A_k\}$ entonces $\{A_i\} \rightarrow \{A_k\}$ es una dependencia transitiva

102

3ª Forma normal

dni	nom_alu	centro	nom_centro	director
1	Olga	EUI	Escuela Universitaria de Informática	Pepe
2	Juana	EUI	Escuela Universitaria de Informática	Pepe
3	Ana	FI	Facultad de Informática	Eva
4	Juan	FI	Facultad de Informática	Eva

CP: {dni}



103

3ª Forma normal

➤ Paso a 3FN:

- Si existe al menos un par de **atributos no-primos** que son **dependientes**.



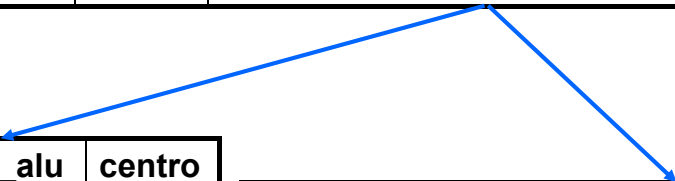
- Sacar el atributo dependiente de la relación y crear una nueva cuya clave primaria será el atributo del que depende.

104

3ª Forma normal

dni	nom_alu	centro	nom_centro	director
1	Olga	EUI	Escuela Universitaria de Informática	Pepe
2	Juana	EUI	Escuela Universitaria de Informática	Pepe
3	Ana	FI	Facultad de Informática	Eva
4	Juan	FI	Facultad de Informática	Eva

CP: {dni}



dni	nom_alu	centro
1	Olga	EUI
2	Juana	EUI
3	Ana	FI
4	Juan	FI

centro	nom_centro	director
EUI	Escuela Universitaria de Informática	Pepe
FI	Facultad de Informática	Eva

105

3ª Forma normal

Alumno(dni, nom_alu, centro, nom_centro, director)

CP: {dni}

→ Alumno(dni, nom_alu, centro)

CP: {dni}

CAj: {centro} → Centro_universitario

→ Centro_universitario(centro, nom_centro, director)

CP: {centro}

106

3ª Forma normal. Ejemplo

Cuenta(numcli: integer, nºcc: cadena(15), tipo_c: cadena(10), saldo: real)

CP: {nºcc}

VNN: {numcli, tipo_c, saldo}

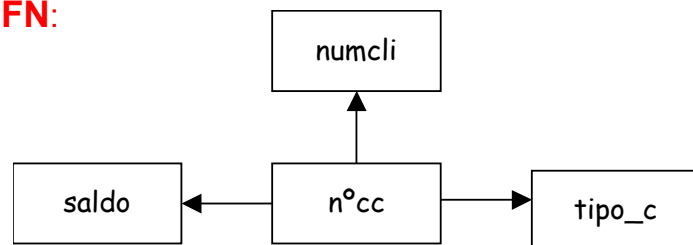
Tarjeta(nºcc:cadena(15),nºtar:cadena(15),tipo_t:cadena(10),comisión:real, capital_límite: real)

CP: {nºtar}

CAj: {nºcc} → Cuenta

VNN: { nºcc, tipo_t, comisión, capital_límite}

En la relación **Cuenta** todos los atributos dependen funcionalmente de la CP y no hay dependencias entre atributos no primos según el diagrama de dependencias funcionales por tanto **está en 3FN**:



107

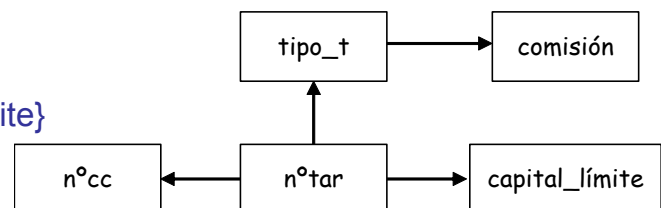
3ª Forma normal. Ejemplo

Tarjeta(nºcc: cadena(15), nºtar: cadena(15), tipo_t: cadena(10), comisión: real, capital_límite: real)

CP: {nºtar}

CAj: {nºcc} → Cuenta

VNN: { nºcc, tipo_t, comisión, capital_límite}



No está en 3FN, existe dependencia entre atributos no primos!!!!!!

Para resolver el problema se divide la relación en dos:

Tarjeta(nºcc: cadena(15), nºtar: cadena(15), tipo_t: cadena(10), capital_límite: real)

CP: {nºtar}

CAj: {nºcc} → Cuenta

VNN: {nºcc, tipo_t, capital_límite}

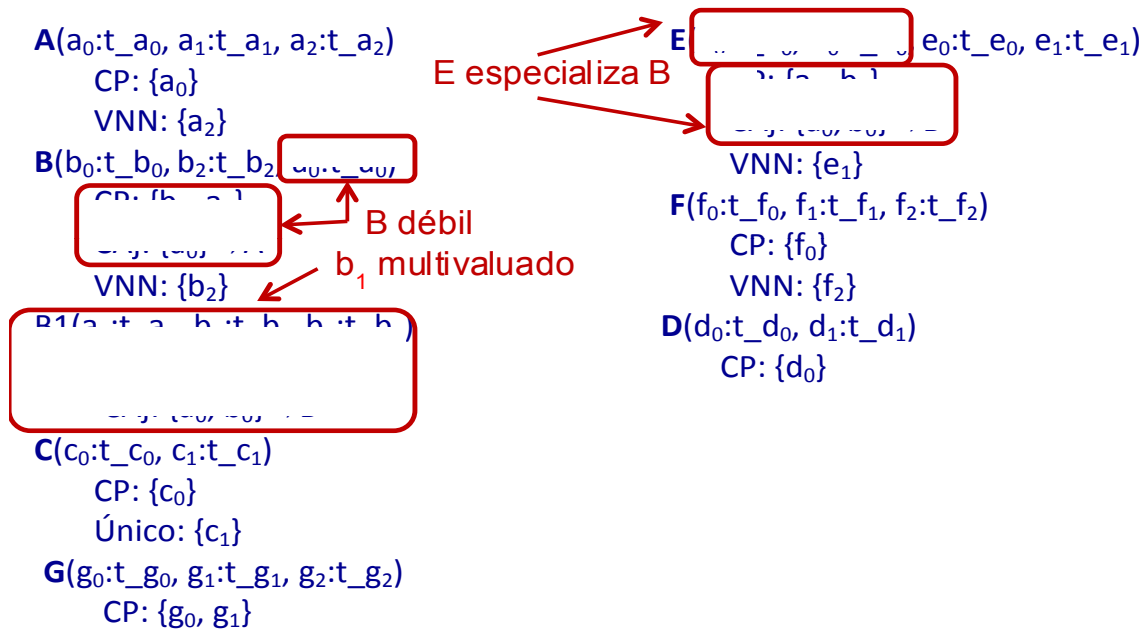
CAj: {tipo_t} → Tipo_Tarjeta

Tipo_Tarjeta (tipo_t: cadena(10), comisión: real)

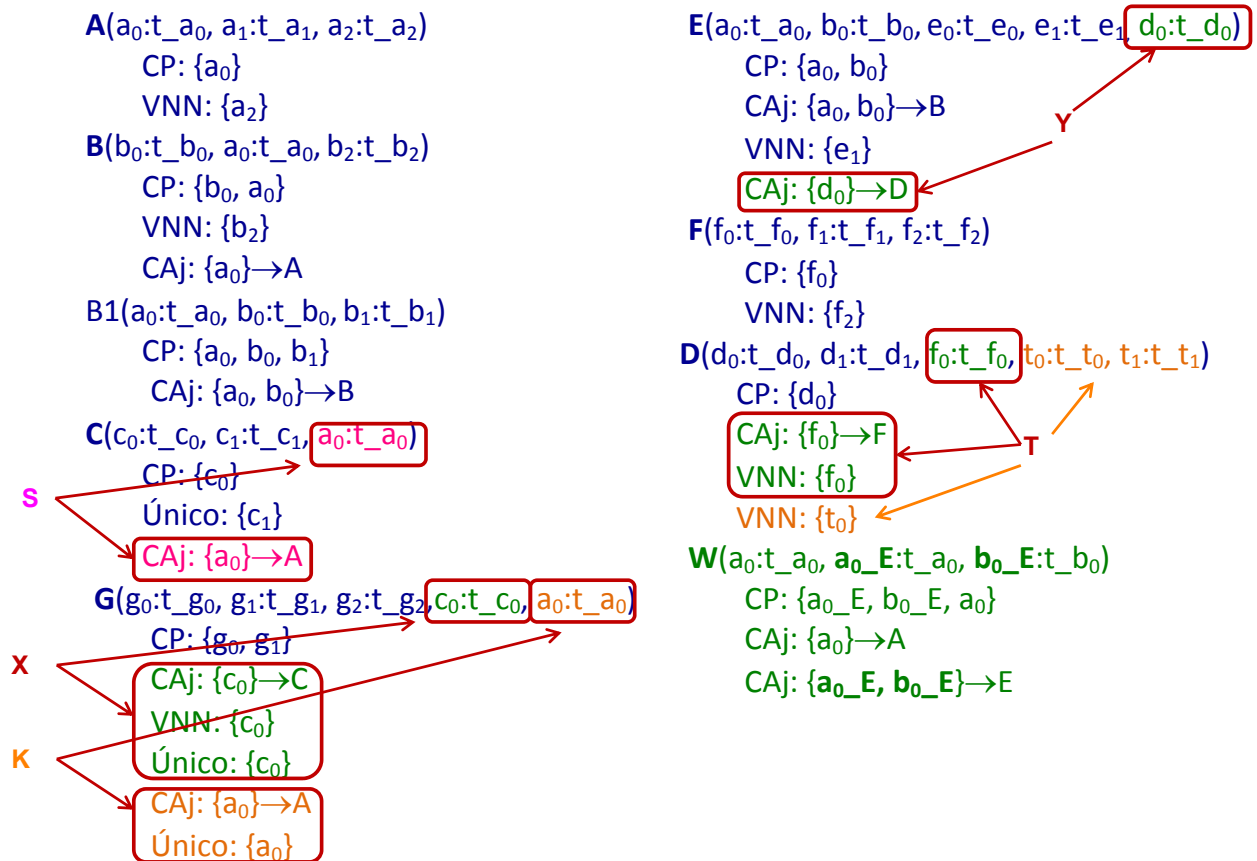
CP: {tipo_t}

VNN: {comisión}

108



113



Restricciones de integridad:

1. Todo valor del atributo a_0 de A aparece al menos una vez en el atributo a_0 de W.

2. Todo valor del atributo a_0 de A aparece al menos una vez en el atributo a_0 de C.

114

Ejercicio

R21 (A: entero, D: texto, G: texto, E: texto, F: texto)

CP: {A}

VNN: {D,G, E, F}

Todo valor de {A} de la relación R21 aparece en R1.

R1(A: entero, B: texto, C: entero)

CP: {A, B}

CAj:{A} -> R21

VNN: {C}

3FN

$\{G\} \rightarrow \{E\}$ $\{G\} \rightarrow \{F\}$ $\{A\} \rightarrow \{D\}$ $\{A\} \rightarrow \{G\}$ $\{A\} \rightarrow \{E\}$ $\{A\} \rightarrow \{F\}$

R1(A: entero, B: texto, C: entero)

CP: {A, B}

CAj:{A} -> R21

VNN: {C}

R21 (A: entero, D: texto, G: texto)

CP: {A}

CAj:{G} -> R22

VNN: {D,G}

R22 (G: entero, E: texto, F: texto)

CP: {G}

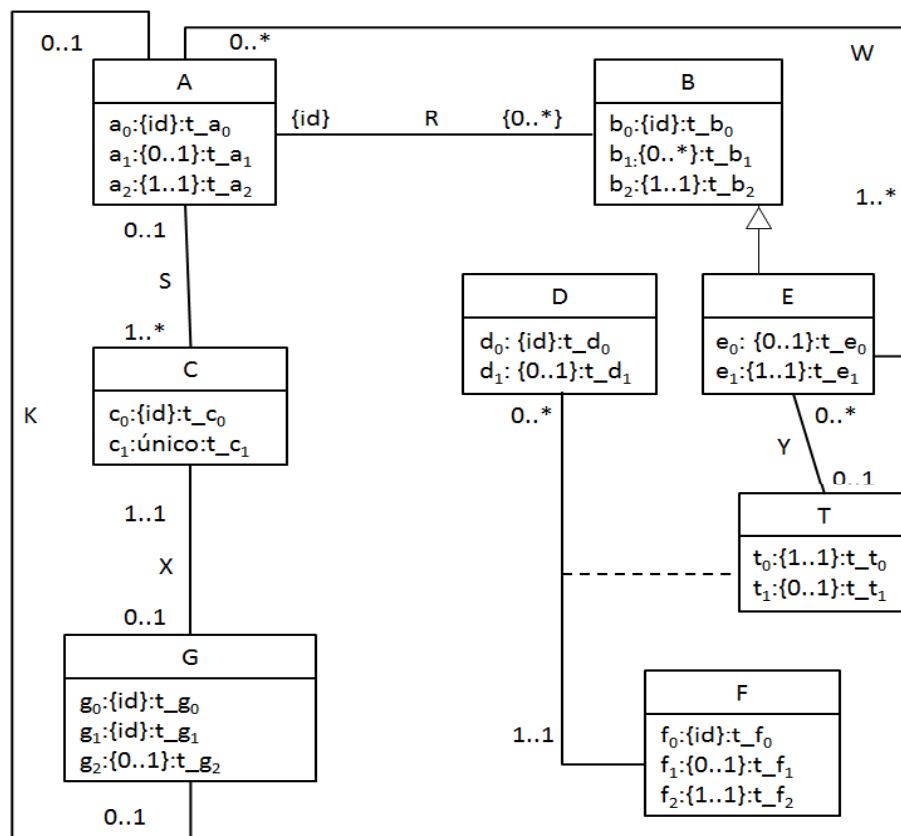
VNN: {E, F}

Todo valor de {A} de la relación R21 aparece en R1.

Todo valor de {G} de la relación R22 aparece en R21.

111

Ejemplo. Diseño lógico



112

Ejercicio

Sea el siguiente esquema de relación:

R(A: entero, B: texto, C: entero, D: texto, E: texto, F: texto, G: texto)

CP: {A, B}

VNN: {C, D, E, F, G}

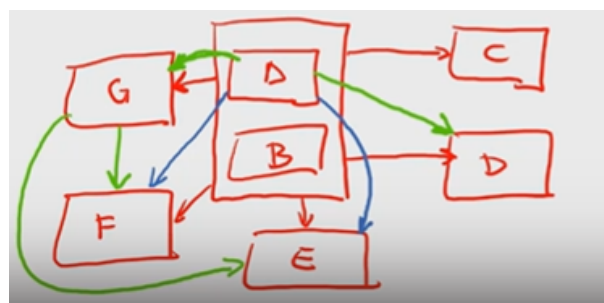
Teniendo en cuenta las dependencias que se exponen a continuación, transfórmala a un conjunto de relaciones en tercera forma normal.

$\{G\} \rightarrow \{E\}$

$\{G\} \rightarrow \{F\}$

$\{A\} \rightarrow \{D\}$

$\{A\} \rightarrow \{G\}$



109

Ejercicio

R(A: entero, B: texto, C: entero, D: texto, E: texto, F: texto, G: texto)

CP: {A, B}

VNN: {C, D, E, F, G}

$\{G\} \rightarrow \{E\}$

$\{G\} \rightarrow \{F\}$

$\{A\} \rightarrow \{D\}$

$\{A\} \rightarrow \{G\}$

Dependencias transitivas:

Si $\{A\} \rightarrow \{G\}$ y $\{G\} \rightarrow \{E\}$

$\{A\} \rightarrow \{E\}$

Si $\{A\} \rightarrow \{G\}$ y $\{G\} \rightarrow \{F\}$

$\{A\} \rightarrow \{F\}$

2FN

$\{G\} \rightarrow \{E\}$

$\{G\} \rightarrow \{F\}$

$\{A\} \rightarrow \{D\}$

$\{A\} \rightarrow \{G\}$

$\{A\} \rightarrow \{E\}$

$\{A\} \rightarrow \{F\}$

R1(A: entero, B: texto, C: entero)

CP: {A, B}

CAj: {A} -> R21

VNN: {C}

R21(A: entero, D: texto, G: texto, E: texto, F: texto)

CP: {A}

VNN: {D, G, E, F}

Todo valor de {A} de la relación R21 aparece en R1.

110

Ejercicio

R21 (A: entero, D: texto, G: texto, E: texto, F: texto)

CP: {A}

VNN: {D,G, E, F}

Todo valor de {A} de la relación R21 aparece en R1.

R1(A: entero, B: texto, C: entero)

CP: {A, B}

CAj:{A} -> R21

VNN: {C}

3FN

$\{G\} \rightarrow \{E\}$ $\{G\} \rightarrow \{F\}$ $\{A\} \rightarrow \{D\}$ $\{A\} \rightarrow \{G\}$ $\{A\} \rightarrow \{E\}$ $\{A\} \rightarrow \{F\}$

R1(A: entero, B: texto, C: entero)

CP: {A, B}

CAj:{A} -> R21

VNN: {C}

R21 (A: entero, D: texto, G: texto)

CP: {A}

CAj:{G} -> R22

VNN: {D,G}

R22 (G: entero, E: texto, F: texto)

CP: {G}

VNN: {E, F}

Todo valor de {A} de la relación R21 aparece en R1.

Todo valor de {G} de la relación R22 aparece en R21.

Normalización

DNI, CodAsg, nombre_alumno, nombre_asignatura
CP {DNI, CodAsg}

R(A: int, B: int, C: char, D: int, E: txt, F: char, G: int, H: char)
 CP: {A, B, C}
 VNN: {D, E, F, G, H}

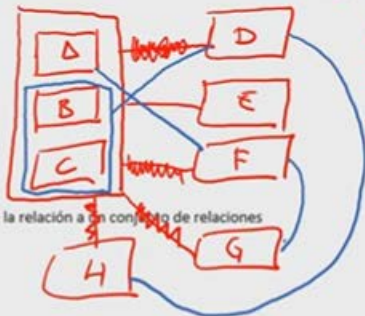
A partir de las dependencias que aparecen a continuación, transforme la relación a un conjunto de relaciones en tercera forma normal

(A) → (F) (B, C) → (D) (F) → (G) (D) → (H)

R(A: int, B: int, C: char, E: txt)
 CP {A, B, C} VNN {E}
 $CA; \{A\} \rightarrow R1(A)$
 $CA; \{B, C\} \rightarrow R2(B, C)$

R1(A: int, F: char, G: int)
 CP {A} VNN {F, G}

R2(B: int, C: char, D: int, H: char)
 CP {B, C} VNN {D, H}



- Todo valor A en R1 debe de aparecer en A de R

- Todo valor B, C de R2 debe aparecer en B, C de R



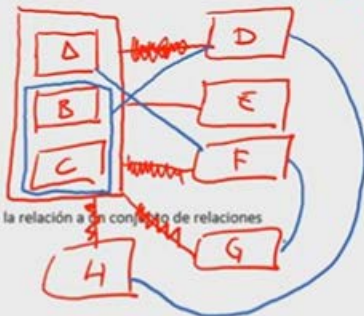
Normalización

DNI, CodAsg, nombre_alumno, nombre_asignatura
 CP { DNI, CodAsg }

R(A: int, B: int, C: char, D: int, E: txt, F: char, G: int, H: char)
 CP: {A, B, C}
 VNN: {D, E, F, G, H}

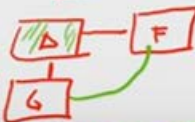
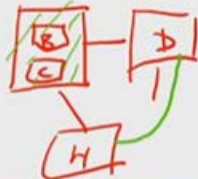
A partir de las dependencias que aparecen a continuación, transforme la relación a un conjunto de relaciones en tercera forma normal

(A) → (F) (B, C) → (D) (F) → (G) (D) → (H)



- Todo valor Δ en $R1$ debe de aparecer en A de R

- Todo valor B, C de $R2$ debe aparecer en B, C de R



$R(A: \text{int}, B: \text{int}, C: \text{char}, E: \text{txt})$
 $CP \{A, B, C\} \quad VNN \{E\}$
 $CA \{A\} \rightarrow R1(A)$
 $CA \{B, C\} \rightarrow R2(B, C)$

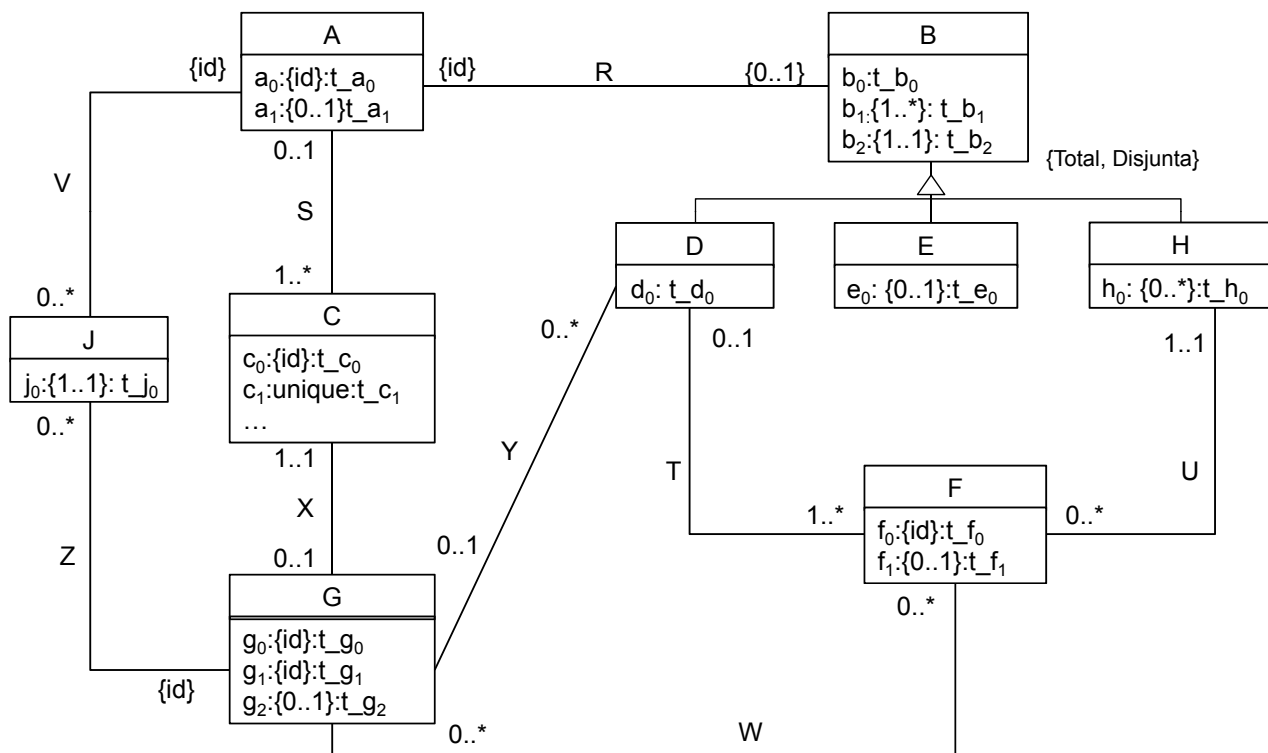
$R1(A: \text{int}, F: \text{char}, G: \text{int})$
 $CP \{A\} \quad VNN \{F, G\}$

$R2(B: \text{int}, C: \text{char}, D: \text{int}, H: \text{char})$

$CP \{B, C\} \quad VNN \{D, H\} \quad CA \{B\} \rightarrow R22(B)$
 $R22(D: \text{int}, H: \text{int}) \quad CP \{D\} \quad VNN \{H\}$

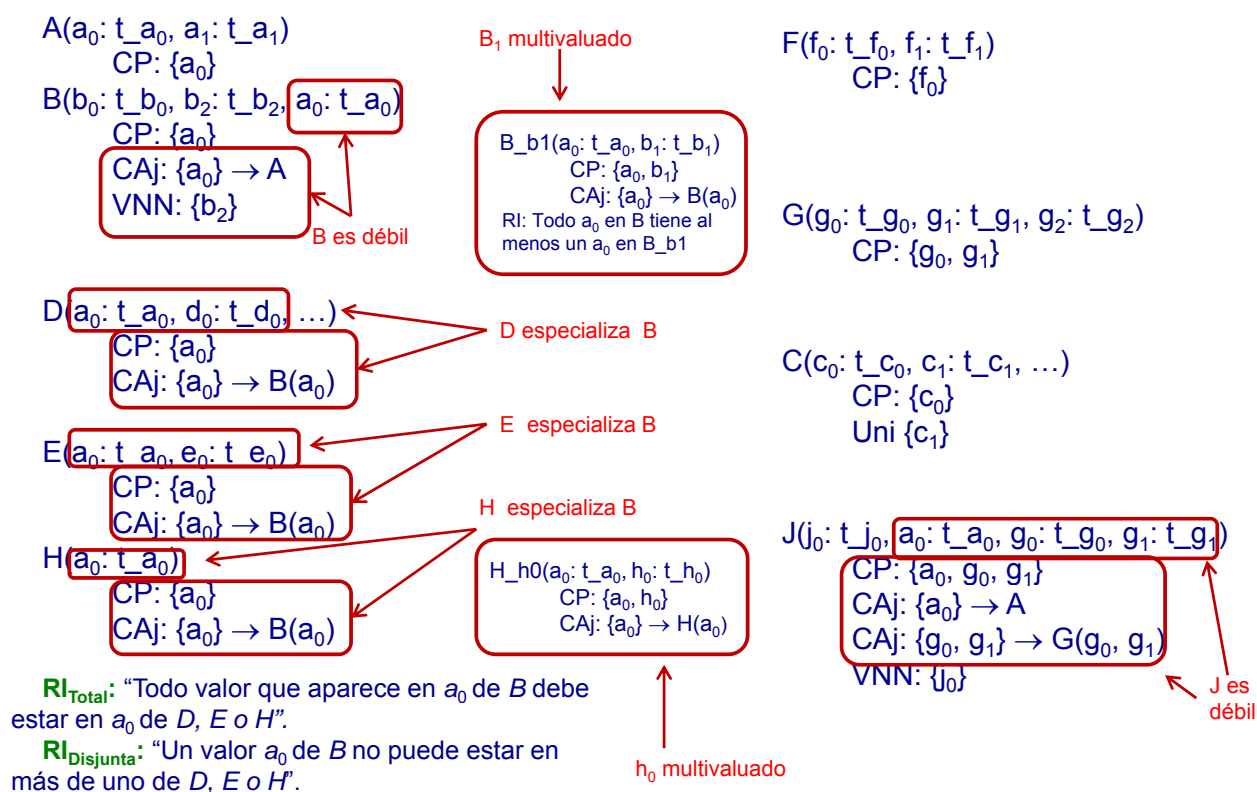
$R11(A: \text{int}, F: \text{char})$
 $CP \{A\} \quad VNN \{F\}$
 $CA \{A\} \rightarrow R12(F)$
 $R12(F: \text{char}, G: \text{int})$
 $CP \{F\} \quad VNN \{G\}$

Ejemplo. Diseño lógico



115

Ejemplo. Transformación de clases



116

A(a₀: t_a₀, a₁: t_a₁)

CP: {a₀}

B(b₀: t_b₀, b₂: t_b₂, a₀: t_a₀)

CP: {a₀}

CAj: {a₀} → A

VNN: {b₂}

D(a₀: t_a₀, d₀: t_d₀, g₀: t_g₀, g₁: t_g₁ ...)

CP: {a₀}

CAj: {a₀} → B(a₀)

CAj: {g₀, g₁} → G(g₀, g₁) Y

E(a₀: t_a₀, e₀: t_e₀)

CP: {a₀}

CAj: {a₀} → B(a₀)

H(a₀: t_a₀)

CP: {a₀}

CAj: {a₀} → B(a₀)

RI_{Total}: "Todo valor que aparece en b₀ de B debe estar en b₀ de D, E o H".

RI_{Disjunta}: "Un valor b₀ de B no puede estar en más de uno de D, E o H".

H_h0(a₀: t_a₀, h₀: t_h₀)

PK: {a₀, h₀}

FK: {a₀} → H(a₀)

B_b1(a₀: t_a₀, b₁: t_b₁)

PK: {a₀, b₁}

FK: {a₀} → B(a₀)

RI: Todo a₀ en B tiene al menos un a₀ en B_b1

F(f₀: t_f₀, f₁: t_f₁, a₀D: t_a₀, a₀H: t_a₀)

CP: {f₀}

CAj: {a₀D} → D(a₀)

RI: "Todo valor a₀ en D debe estar en F" T U

CP: {a₀H} → H(a₀) VNN: {a₀H}

G(g₀: t_g₀, g₁: t_g₁, g₂: t_g₂, c₀: t_c₀)

CP: {g₀, g₁}

UNI: {c₀} NNV: {c₀}

CAj: {c₀} → C(c₀) X

C(c₀: t_c₀, c₁: t_c₁, ..., a₀: t_a₀)

CP: {c₀}

UNI: {c₁}

CAj: {a₀} → A S

RI: "Todo valor a₀ en A debe estar en C"

J(j₀: t_j₀, a₀: t_a₀, g₀: t_g₀, g₁: t_g₁)

CP: {a₀, g₀, g₁}

CAj: {a₀} → A

CAj: {g₀, g₁} → G(g₀, g₁)

VNN: {j₀}

W(g₀: t_g₀, g₁: t_g₁, f₀: t_f₀)

CP: {g₀, g₁, f₀}

CAj: {g₀, g₁} → G(g₀, g₁)

CAj: {f₀} → F