Tema 4: Diseño en BBDD Relacionales

- Diseño lógico
- Dependencias funcionales y normalización
- Diseño físico

Ignacio J. Blanco, Jesús Campaña, Carlos Cruz, María Ros, Daniel Sánchez

1 Forma Normal – 1FN

Forma no normalizada (UNF)	Una tabla que contiene uno o más grupos repetitivos.
Primera forma normal (1NF)	Una relación en la que la intersección de toda fila y columna contiene un valor y sólo un valor.

1FN- Técnicas

(1) *Introduciendo datos apropiados en las columnas vacías de las filas que contienen los datos repetitivos*. En otras palabras, llenamos los blancos duplicando los datos no repetitivos siempre que sea necesario. Esta técnica se suele denominar 'aplanamiento' de la tabla.

(2) *Colocando los datos repetitivos, junto con una copia de los atributos clave originales en una relación independiente*. Algunas veces la tabla no normalizada puede contener más de un grupo repetitivo o puede contener grupos repetitivos dentro de otros grupos repetitivos. En tales casos, se aplica esta técnica de forma sucesiva hasta que no quede ningún grupo repetitivo. Un conjunto de relaciones está en la forma 1NF si no contiene ningún grupo repetitivo.

1FN- Técnicas

Contrato de alquiler de *DreamHome*

Contrato de alquiler de *DreamHome*

Contrato de alquiler de *DreamHome*

Contrato de alquiler de *DreamHome*

Número de cliente (introducir si se conoce) CR76 Nombre completo (introducir) John Kay	Número de inmueble PG4 Dirección del inmueble 6 Lawrence St, Glasgow
Alquiler mensual 350 Inicio alquiler 01/07/03 Fin alquiler 31/08/04	Número de propietario (introducir si se conoce) CO40 Nombre completo (introducir) Tina Murphy

1FN- Técnicas

ClientRental

clientNo	cName	propertyNo	pAddress	rentStart	rentFinish	rent	ownerNo	oName
CR76	John Kay	PG4 PG16	6 Lawrence St, Glasgow 5 Novar Dr, Glasgow	1-Jul-03 1-Sep-04	31-Aug-04 1-Sep-05	350 450	CO40 CO93	Tina Murphy Tony Shaw
CR56	Aline Stewart	PG4 PG36 PG16	6 Lawrence St, Glasgow 2 Manor Rd, Glasgow 5 Novar Dr, Glasgow	1-Sep-02 10-Oct-03 1-Nov-05	10-June-03 1-Dec-04 10-Aug-06	350 375 450	CO40 CO93 CO93	Tina Murphy Tony Shaw Tony Shaw

Tabla UNF

1FN- Técnicas

ClientRental

clientNo	propertyNo	cName	pAddress	rentStart	rentFinish	rent	ownerNo	oName
CR76	PG4	John Kay	6 Lawrence St, Glasgow	1-Jul-03	31-Aug-04	350	CO40	Tina Murphy
CR76	PG16	John Kay	5 Novar Dr, Glasgow	1-Sep-04	1-Sep-05	450	CO93	Tony Shaw
CR56	PG4	Aline Stewart	6 Lawrence St, Glasgow	1-Sep-02	10-Jun-03	350	CO40	Tina Murphy
CR56	PG36	Aline Stewart	2 Manor Rd, Glasgow	10-Oct-03	1-Dec-04	375	CO93	Tony Shaw
CR56	PG16	Aline Stewart	5 Novar Dr, Glasgow	1-Nov-05	10-Aug-06	450	CO93	Tony Shaw

Tabla 1FN

1FN- Técnicas

Client

clientNo	cName
CR76	John Kay
CR56	Aline Stewart

PropertyRentalOwner

clientNo	propertyNo	pAddress	rentStart	rentFinish	rent	ownerNo	oName
CR76	PG4	6 Lawrence St, Glasgow	1-Jul-03	31-Aug-04	350	CO40	Tina Murphy
CR76	PG16	5 Novar Dr, Glasgow	1-Sep-04	1-Sep-05	450	CO93	Tony Shaw
CR56	PG4	6 Lawrence St, Glasgow	1-Sep-02	10-Jun-03	350	CO40	Tina Murphy
CR56	PG36	2 Manor Rd, Glasgow	10-Oct-03	1-Dec-04	375	CO93	Tony Shaw
CR56	PG16	5 Novar Dr, Glasgow	1-Nov-05	10-Aug-06	450	CO93	Tony Shaw

Tabla 1FN

Dependencias funcionales

- Dada una relación *R* con n atributos y dos subconjuntos de atributos de *R* llamados α y β, se dice que α determina funcionalmente a β o que β depende funcionalmente de α si para cualquier pareja de tuplas *s* y *t* de la instancia *r* de la relación, que tengan iguales valores para los atributos del subconjunto α se verifica que tienen los mismos valores para los atributos del subconjunto β.

$$R(A_{1,} A_{2,\ldots}, A_n), \alpha \subset R, \beta \subset R, \alpha \rightarrow\rightarrow \beta \; sii$$
$$\alpha \rightarrow \beta \wedge \nexists \gamma \subset \alpha | \gamma \rightarrow \beta$$

Dependencias funcionales

B es funcionalmente dependiente de **A**
A determina funcionalmente a **B**

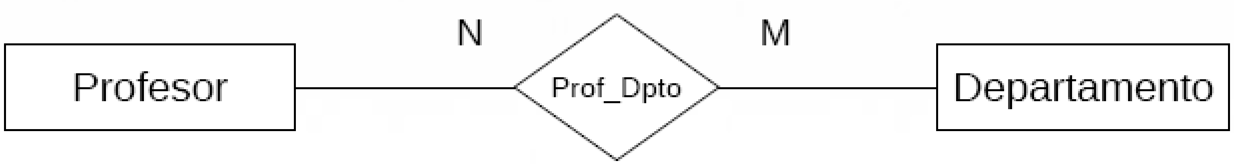
Dependencias funcionales

- Son ***restricciones de integridad*** que permiten conocer que ***interrelaciones*** existen entre dos o más atributos del mundo real.
- Son propiedades ***inherentes*** al ***contenido semántico*** de los datos, que se han de cumplir para cualquier extensión del esquema de relación.
- Informalmente, Y **depende funcionalmente** de X si ***cada valor*** de X tiene asociado ***siempre el mismo*** valor de Y en una relación R que contiene a X y Y como atributos.

Dependencias funcionales

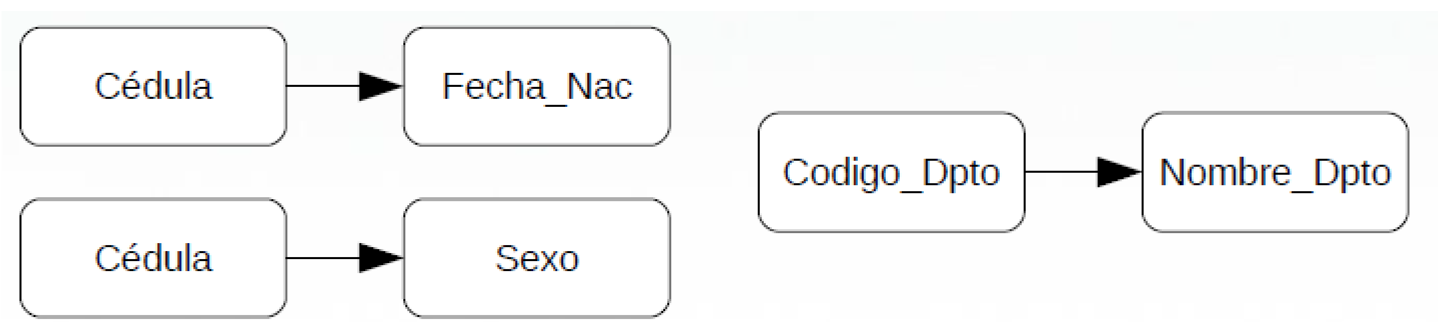
El resultado de una consulta cualquiera
(por ejemplo, de un producto entre la tabla profesor y departamento):

Cédula	Fecha_Nac	Sexo	Código_Depto	Nombre_Depto
9.980.623	06/01/73	M	01	Computación
10.334.890	**06/01/76**	**F**	01	Computación
17.544.672	**06/01/84**	**M**	03	Investigación
12.334.222	**06/01/77**	M	02	Control
13.566.002	**12/01/78**	**F**	02	Control
10.334.890	**06/01/76**	**F**	02	Control
12.334.222	**06/01/77**	**M**	01	Computación
13.434.122	06/01/78	F	03	Investigación
13.566.002	**12/01/78**	**F**	03	Investigación
17.544.672	**06/01/84**	**M**	02	Control
18.244.670	06/01/85	M	01	Computación



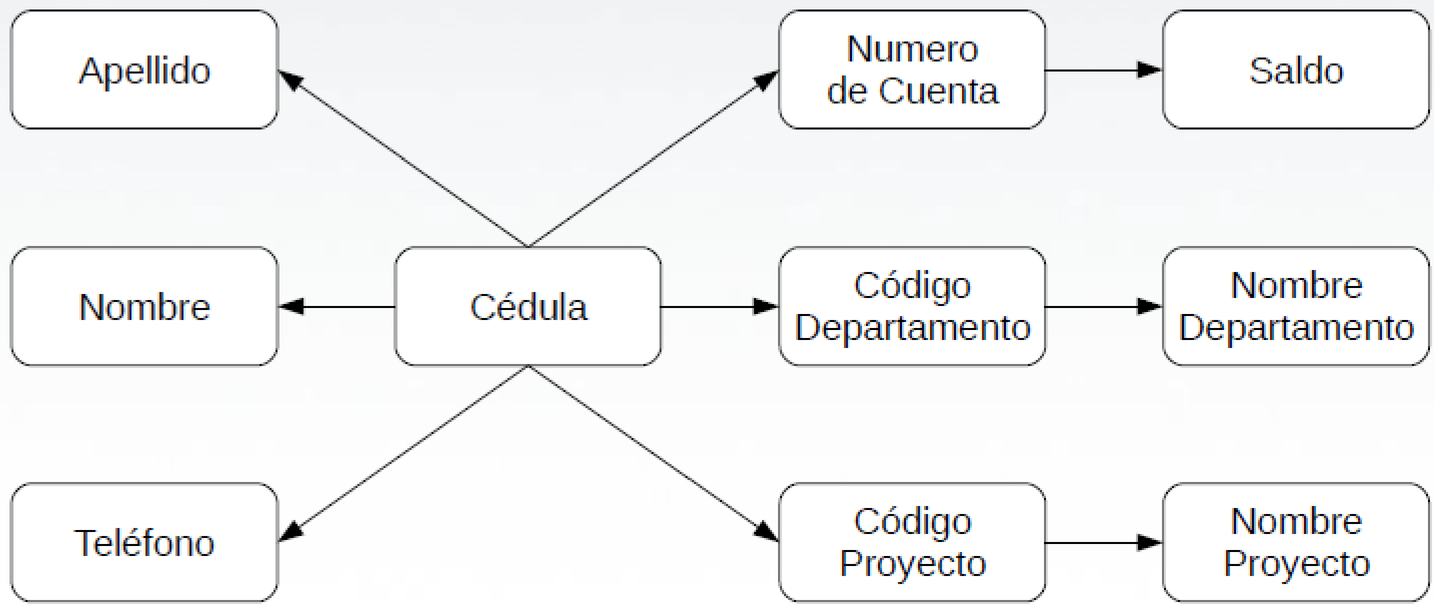
Dependencias funcionales

Cédula	Fecha_Nac	Sexo	Código_Depto	Nombre_Depto
9.980.623	06/01/73	M	01	Computación
10.334.890	06/01/76	F	01	Computación
17.544.672	06/01/84	M	03	Investigación
12.334.222	06/01/77	M	02	Control
13.566.002	12/01/78	F	02	Control
10.334.890	06/01/76	F	02	Control
12.334.222	06/01/77	M	01	Computación
13.434.122	06/01/78	F	03	Investigación
13.566.002	12/01/78	F	03	Investigación
17.544.672	06/01/84	M	02	Control
18.244.670	06/01/85	M	01	Computación



Dependencias funcionales

Son restricciones de integridad que permiten conocer que interrelaciones existen entre los atributos del mundo real



Normalización: Formas Normales

Forma Normal	Año	Autor	Basada en...
1FN	1970	Codd	No basada en dependencias: Impone dominios atómicos
2FN	1970	Codd	Dependencias funcionales
3FN	1970	Codd	
FNBC	1972	Boyce & Codd	
4FN	1977	Fagin	Dependencias multivaluadas
5FN	1979	Rissanen	Dependencias de reunión

Ignacio J. Blanco, Jesús Campaña, Carlos Cruz, María Ros, Daniel Sánchez

Dependencias funcionales

- Dependencia funcional completa:
  - Dada una relación *R* con n atributos y dos subconjuntos de atributos de *R* llamados α y β, se dice que α *determina funcionalmente* a β *de forma completa* o *determina completamente a* β si α determina funcionalmente a β y no hay ningún subconjunto de atributos de α que determine funcionalmente a β.

Dependencias funcionales

Si A y B son atributos de la relación R,
B depende funcionalmente de manera completa de A si B depende funcionalmente de A; pero no de ningún subconjunto propio de A.

O sea, una dependencia funcional A⟶B es una dependencia funcional completa si la eliminación de cualquier atributo de A hace que la dependencia deje de existir.

Dependencias funcionales

- Atributo primo:
  - Se llama así a aquel atributo que forma parte de una clave candidata.

2da. Forma Normal – 2FN

Decimos que una relación *R* está en segunda forma normal (2FN) sii:

- Está en primera forma normal (1NF) y
- Todos sus atributos no primos dependen de forma completa de las claves candidatas.

- Un buen diseño conceptual genera tablas que están en segunda forma normal.

2da. Forma Normal – 2FN

- Se basa en el concepto de **dependencia funcional total**, una dependencia funcional $X \rightarrow Y$ es total si la eliminación de cualquier atributo A de X rompe la dependencia.
- Una relación está en 2FN si está en 1FN y todo atributo que no sea clave depende de forma total de la clave.
- La 2FN permite eliminar las ***redundancias*** para que ningún atributo sea determinado ***sólo por una parte*** de una clave.

2da. Forma Normal – 2FN

Ejemplo:

STOCK (<u>#almacén, #producto</u>, cantidad, dirección_almacén)

2da. Forma Normal – 2FN

Ejemplo:

STOCK (<u>#almacén, #producto</u>, cantidad, dirección_almacén)

- Problemas
  - La dirección del almacén se repite para cada producto existente en el inventario del almacén.
  - Si la dirección del almacén cambia, hay que actualizar todas las tuplas relativas a los distintos productos almacenados en el almacén.
  - Debido a la redundancia existente, pueden aparecer inconsistencias si distintas tuplas contienen distintos valores para la dirección de un mismo almacén.
  - Si en algún momento no existiese stock alguno en el almacén, no habría ningún sitio donde almacenar su dirección.

2da. Forma Normal – 2FN

Ejemplo:

STOCK (<u>#almacén, #producto</u>, cantidad, dirección_almacén)

- Causa
  - La clave de la relación es una clave compuesta:

    {#almacén, #producto}
  - El atributo 'dirección_almacén' no pertenece a la clave y depende sólo de parte de ella (del atributo '#almacén').

La relación STOCK no está en 2FN

2da. Forma Normal – 2FN

- Ejemplo:

$$DF = \left\{ \begin{array}{c} \{almacén, producto\} \rightarrow cantidad, \{almacén, producto\} \rightarrow direccionalmacen, \\ almacén \rightarrow direccionalmacen \end{array} \right\}$$

2da. Forma Normal – 2FN

- Teorema de Heath:
  - Sea R una relación con atributos A, B, y C, donde se verifica $B \rightarrow C$. Entonces, la descomposición de R en dos relaciones:
    - $R_1(A,B)$
    - $R_2(B,C)$

    es una *descomposición sin pérdidas*

2da. Forma Normal – 2FN

- Descomposición sin pérdidas:
  - Sea (R,r) una relación que se descompone en (R1,r1) y (R2,r2), se dice que la descomposición es sin pérdidas sii:

$$R_1 \cup R_2 = R$$
$$r_1 \; JOIN \; r_2 = r$$

2da. Forma Normal – 2FN

- En nuestro ejemplo, dependencia entre #almacén (parte de la clave) y dirección_almacén, hace que la relación no esté en segunda forma normal.
- Aplicamos el Teorema de Heath sobre esa dependencia y nos quedan dos relaciones.

2da. Forma Normal – 2FN

- Ejemplo (descomposición sin pérdidas):

$$R_1 = \{almacén\,,\ producto\,,\ cantidad\}$$
$$DF_1 = \{\{almacén\,,\ producto\} \rightarrow cantidad\}$$
$$R_2 = \{almacén\,,\ direcciónalmacén\}$$
$$DF_2 = \{almacén \rightarrow direcciónalmacén\}$$

2da. Forma Normal – 2FN

- No obstante, la relación R cumplía una serie de restricciones (dependencias funcionales) antes de descomponerse. ¿Esas restricciones se siguen cumpliendo en las dos relaciones resultantes?:
  - En nuestro ejemplo, todas las dependencias originales se encuentran dentro de DF_1 o de DF_2 excepto {#almacén, producto}$\rightarrow$dirección_almacén que parece haber se perdido.
  - ¿Se ha perdido realmente?

Ignacio J. Blanco, Jesús Campaña, Carlos Cruz, María Ros, Daniel Sánchez

2da. Forma Normal – 2FN

- ¿Se ha perdido realmente?
  - La respuesta no es tan sencilla porque, el hecho de la definición de dependencia funcional hace que se deriven una serie de axiomas y reglas que nos permiten operar con ellas. Se conocen como los **Axiomas de Armstrong**.
  - A lo mejor, podemos recuperar lo que supuestamente se ha perdido a partir de las dependencias que quedan.

Axiomas de Armstrong

Inferir nuevas dependencias funcionales a partir de otras

(1) **Reflexividad:** Si B es un subconjunto de A, entonces $A \rightarrow B$

(2) **Aumentación:** Si $A \rightarrow B$, entonces $A,C \rightarrow B,C$

(3) **Transitividad:** Si $A \rightarrow B$ y $B \rightarrow C$, entonces $A \rightarrow C$

Reglas de Armstrong

Inferir nuevas dependencias funcionales a partir de otras

(4) **Autodeterminación:** $A \rightarrow A$

(5) **Descomposición:** Si $A \rightarrow B,C$, entonces $A \rightarrow B$ y $A \rightarrow C$

(6) **Unión:** Si $A \rightarrow B$ y $A \rightarrow C$, entonces $A \rightarrow B,C$

(7) **Composición:** Si $A \rightarrow B$ y $C \rightarrow D$, entonces $A,C \rightarrow B,D$

Ignacio J. Blanco, Jesús Campaña, Carlos Cruz, María Ros, Daniel Sánchez

Preservación de dependencias

- Volver a obtener dependencias que parecen haberse perdido, es un proceso tedioso y requiere claridad de visión, ya que se pueden aplicar secuencias de axiomas y reglas que pueden llevarnos a callejones sin salida.
- Para verificar si las dependencias siguen existiendo sin que estén explícitamente presentes (es decir, que se pueden deducir de otras) se emplean otros conceptos.

Preservación de dependencias

Cierre de un conjunto de dependencias funcionales F:

- Se nota por F^+ y representa el conjunto de todas las dependencias funcionales que pueden deducirse de las dependencias funcionales de F aplicando los Axiomas y las Reglas de Armstrong en una secuencia finita de pasos.

Preservación de dependencias

Ejemplo del cálculo de F^+ usando los axiomas de Armstrong

Sea R=(A,B,C,D) y F={A→C,B→D}

Aplicación sucesiva de Axioma 1:

A →A, B →B, C →C, D →D AC → AC, AC →C, AC →A, AB →AB, AB →B, AB →A,BC →BC, BC →B, BC →C, AD → AD, AD →D, AD →A, BD →BD, BD →B, BD →D, DC →D, DC →C, ABC →ABC, ABC →AB, ABC →AC, ABC →BC, ABD→ABD, ABD →AB, ABC → A, ABC → B, ABC → C, ABD →AD, ABD →BD, BCD →BCD, BCD →BC, ABD → A, ABD → B, ABD →D, BCD →BCD, BCD →BD,BCD →BC, BCD →CD, BCD → C, BCD → B, BCD →D, ABCD →ABCD, ABCD →ABC, ABCD →ABD, ABCD →BCD, ABCD →AB, ABCD →AC, ABCD →AD, ABCD →BC, ABCD →BD, ABCD →CD, ABCD →A, ABCD →B, ABCD →C, ABCD →D

Aplicación sucesiva de Axioma 2:

A→C⇒{AB→BC , AB→ABC, ABD→BCD, AD→DC, AD→ADC, ADB→ABDC} ; B→D ⇒{AB→AD, AB→ABD, ABC→ADC, BC→DC, BC→BDC, ABC→ABDC}

…

Aplicación sucesiva de Axioma 4:

AB→C, ABD→C, AD→C, AB→D, ABC→D ....

Preservación de dependencias

Problema:

El cálculo de F^+ es muy tedioso, incluso si se aplican los axiomas de Armstrong.

Se pueden evitar las dependencias triviales, salvo para deducir otras.

Sería deseable trabajar con conjuntos de dependencias lo más simples posibles. Por ejemplo:

$\{A \rightarrow BCD, A \rightarrow BC, A \rightarrow D, A \rightarrow CD\}$
nos dice lo mismo que $\{A \rightarrow B, A \rightarrow C, A \rightarrow D\}$

En cualquier caso si $F \supseteq \{A \rightarrow B_1, ..., A \rightarrow B_n\}$,
entonces F^+ incluye al menos $\mathbf{2^n}$ dependencias funcionales.

Solución: Trabajar con cierres "parciales" de subconjuntos de atributos con respecto a un conjunto de dependencias funcionales.

Preservación de dependencias

Cierre de un conjunto de atributos α en base a un conjunto de dependencias funcionales F:

- Se nota por α^+ y representa el conjunto de todos los atributos que son determinados por los atributos de α en conjunto mediante dependencias de F^+
- Son equivalentes:

$$\alpha \rightarrow A \in F^+$$
$$A \notin \alpha_F^+ \; implica\ que\ \alpha \rightarrow A \notin F^+$$

Preservación de dependencias

Cálculo de α^+:

- Inicializamos $\alpha^+=\alpha$
- Mientras que α^+ cambie
  - Si $\beta \rightarrow \gamma \in F$ y $\beta \in \alpha^+ \Rightarrow \gamma \in \alpha^+$

Preservación de dependencias

Cálculo de α^+ (un ejemplo):

- R(A,B,C,D,E,F)
- F={$AB \rightarrow C$, $D \rightarrow EF$, $C \rightarrow A$, $BE \rightarrow C$, $BC \rightarrow D$, $CF \rightarrow BD$, $ACD \rightarrow B$, $CE \rightarrow AF$}
- ¿BC^+?

Preservación de dependencias

Cálculo de α^+ (un ejemplo):

$BC^+=\{B, C\}$
$BC^+=\{B, C, A\}$ por $C \rightarrow A$
$BC^+=\{B, C, A, D\}$ por $BC \rightarrow D$
$BC^+=\{B, C, A, D, E, F\}$ por $D \rightarrow EF$

$F=\{AB \rightarrow C, D \rightarrow EF, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CF \rightarrow BD, ACD \rightarrow B, CE \rightarrow AF\}$

Preservación de dependencias

- Volviendo a nuestro ejemplo:
  - Hemos partido la relación en dos relaciones y sus conjuntos de dependencias funcionales en otros dos conjuntos.
  - Por el Teorema de Heath, se verifica que la descomposición es sin pérdidas, es decir, que la reunión natural de R_1 y R_2 tiene todos los datos pero ¿qué dependencias observa esa reunión?

$$R_1 = \{almacén, producto, cantidad\}$$
$$DF_1 = \{\{almacén, producto\} \rightarrow cantidad\}$$
$$R_2 = \{almacén, direcciónalmacén\}$$
$$DF_2 = \{almacén \rightarrow direcciónalmacén\}$$

¿ *almacén→direccionalmacen* ?

Preservación de dependencias

- Parece lógico pensar que observa todas las dependencias que hay en DF_1 y DF_2, ...
- y todas las que se puedan deducir de ellas, es decir, $(DF_1 \cup DF_2)^+$
- Entonces, ¿para saber si hemos perdido *{#almacén, producto}→dirección_almacén* hemos de calcular todo $(DF_1 \cup DF_2)^+$?

Preservación de dependencias

- No es necesario, sino que basta con comprobar si *{#almacén, producto}*$\rightarrow$*dirección_almacén* pertenece a $(DF_1 \cup DF_2)^+$
- Y eso es fácil, porque basta con comprobar si dirección_almacén pertenece al cierre de atributos *{#almacén, producto}*$^+_{DF_1 \cup DF_2}$

Preservación de dependencias

- Calculando:

{almacén, producto}$^+$={almacén, producto}
{almacén, producto}$^+$={almacén, producto, cantidad} por {almacén, producto}$\rightarrow$cantidad
{almacén, producto}$^+$={almacén, producto, cantidad, dirección_almacén} por almacén$\rightarrow$dirección_almacén

- Dado que *dirección_almacén* está en el cierre de atributos, se puede deducir de ellos y la dependencia existe, por lo que no se ha perdido.

$$R_1=\{almacén, producto, cantidad\}$$
$$DF_1=\{\{almacén, producto\}\rightarrow cantidad\}$$
$$R_2=\{almacén, direcciónalmacén\}$$
$$DF_2=\{almacén\rightarrow direcciónalmacén\}$$

Preservación de dependencias

- Como se ha visto, trabajar con un conjunto de dependencias F y su cierre F+ puede ser tedioso.
- De hecho, F puede contener dependencias que son redundantes, es decir, que pueden obtenerse a partir de otras dependencias presentes en el mismo conjunto.
- De ahí, que se suela trabajar con un conjunto simplificado de dependencias funcionales.

Recubrimiento minimal de dependencias

- Se llama **recubrimiento minimal o canónico** de un conjunto de dependencias funcionales *F* y se nota por *F'* al conjunto que cumple:

  $$F^{+} = (F')^{+}$$

  es decir, que cualquier dependencia que se puede obtener a través de *F* se puede obtener a través de *F'*, pero *F'* está formada por dependencias con estructura mucho más simple que las de *F*.

Recubrimiento minimal de dependencias

- El proceso de obtención del recubrimiento minimal consiste en partir de F para simplificar las dependencias, simplificando:
  - La parte derecha de la dependencia
  - La parte izquierda de la dependencia
  - La dependencia en sí
- La obtención de F' se basa en un algoritmo con tres pasos.

Recubrimiento minimal de dependencias

- Lo explicaremos con el mismo ejemplo anterior:
  - R(A,B,C,D,E,F)
  - F={AB→C, D→EF, C→A, BE→C, BC→D, CF→BD, ACD→B, CE→AF}

Recubrimiento minimal de dependencias

- Paso 1: obtención de $F^{(1)}$ mediante aplicación de la regla de descomposición a todas las dependencias que tengan parte derecha compuesta.

  $F^{(1)}=\{AB\rightarrow C, D\rightarrow E, D\rightarrow F, C\rightarrow A, BE\rightarrow C, BC\rightarrow D, CF\rightarrow B, CF\rightarrow D, ACD\rightarrow B, CE\rightarrow A, CE\rightarrow F\}$

- Se ve claramente que si aplicamos la regla de unión sobre cada pareja de dependencias en rojo, volvemos a obtener la original.

$F=\{AB\rightarrow C, D\rightarrow EF, C\rightarrow A, BE\rightarrow C, BC\rightarrow D, CF\rightarrow BD, ACD\rightarrow B, CE\rightarrow AF\}$

Ignacio J. Blanco, Jesús Campaña, Carlos Cruz, María Ros, Daniel Sánchez

Recubrimiento minimal de dependencias

- Paso 2: obtención de $F^{(2)}$ mediante simplificación de la parte izquierda de las dependencias eliminando *atributos raros*.
- Sea una dependencia con la parte izquierda compuesta de la forma $\alpha A \rightarrow B$, se dice que *A es raro con respecto a* α sii $A \in \alpha^+$, es decir, que A depende funcionalmente de los atributos que le acompañan.
- Cada atributo raro que aparezca con respecto a los que le acompañan, se suprime.

Ignacio J. Blanco, Jesús Campaña, Carlos Cruz, María Ros, Daniel Sánchez

Recubrimiento minimal de dependencias

- Paso 2:
  - $AB \rightarrow C$,
    - $A^+=\{A\}$, B no pertenece a A^+ luego B no es raro con respecto a A
    - $B^+=\{B\}$, A no pertenece a B^+ luego A no es raro con respecto a B

  luego $AB \rightarrow C$ se queda como está.

$F^{(1)}=\{AB \rightarrow C, D \rightarrow E, D \rightarrow F, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CF \rightarrow B, CF \rightarrow D, ACD \rightarrow B, CE \rightarrow A, CE \rightarrow F\}$

Recubrimiento minimal de dependencias

- Paso 2: ...
  - $BE \rightarrow C$,
    - $B^+=\{B\}$, E no pertenece a B^+ luego E no es raro con respecto a B
    - $E^+=\{E\}$, B no pertenece a E^+ luego B no es raro con respecto a E

  luego $BE \rightarrow C$ se queda como está.

$F^{(1)}=\{AB \rightarrow C,\ D \rightarrow E,\ D \rightarrow F,\ C \rightarrow A,\ BE \rightarrow C,\ BC \rightarrow D,\ CF \rightarrow B,\ CF \rightarrow D,\ ACD \rightarrow B,\ CE \rightarrow A,\ CE \rightarrow F\}$

Recubrimiento minimal de dependencias

- Paso 2: ...
  - $BC \rightarrow D$,
    - $B^+=\{B\}$, C no pertenece a B^+ luego C no es raro con respecto a B
    - $C^+=\{C, A\}$, B no pertenece a C^+ luego B no es raro con respecto a C

  luego $BC \rightarrow D$ se queda como está.

$F^{(1)}=\{AB \rightarrow C, D \rightarrow E, D \rightarrow F, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CF \rightarrow B, CF \rightarrow D, ACD \rightarrow B, CE \rightarrow A, CE \rightarrow F\}$

Recubrimiento minimal de dependencias

- Paso 2: ...
  - $CF \rightarrow B$, $CF \rightarrow D$,
    - $F^+=\{F\}$, C no pertenece a F^+ luego C no es raro con respecto a F
    - $C^+=\{C, A\}$, F no pertenece a C^+ luego F no es raro con respecto a C

  luego $CF \rightarrow B$, $CF \rightarrow D$ se quedan como están.

$F^{(1)}=\{AB \rightarrow C, D \rightarrow E, D \rightarrow F, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CF \rightarrow B, CF \rightarrow D, ACD \rightarrow B, CE \rightarrow A, CE \rightarrow F\}$

Ignacio J. Blanco, Jesús Campaña, Carlos Cruz, María Ros, Daniel Sánchez

Recubrimiento minimal de dependencias

- Paso 2: ...
  - $ACD \rightarrow B$,
    - $\{AC\}^+=\{A, C\}$, D no pertenece a $\{AC\}^+$ luego D no es raro con respecto a {AC}
    - $\{AD\}^+=\{A, D, E, F\}$, C no pertenece a $\{AD\}^+$ luego C no es raro con respecto a {AD}
    - $\{CD\}^+=\{C, D, E, F, A, B\}$, A pertenece a $\{CD\}^+$ luego **A es raro con respecto a {CD}**

  luego $ACD \rightarrow B$ se cambia por $CD \rightarrow B$, pero hay que seguir comprobando dentro de {CD}

$F^{(1)}=\{AB \rightarrow C, D \rightarrow E, D \rightarrow F, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CF \rightarrow B, CF \rightarrow D, ACD \rightarrow B, CE \rightarrow A, CE \rightarrow F\}$

Recubrimiento minimal de dependencias

- Paso 2: ...
  - $CD \rightarrow B$,
    - $C^+=\{C, A\}$, D no pertenece a C^+ luego D no es raro con respecto a C
    - $D^+=\{D, E, F\}$, C no pertenece a D^+ luego C no es raro con respecto a D

  luego $CD \rightarrow B$ queda como está.

$F^{(1)}=\{AB \rightarrow C, D \rightarrow E, D \rightarrow F, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CF \rightarrow B, CF \rightarrow D, ACD \rightarrow B, CE \rightarrow A, CE \rightarrow F\}$

Recubrimiento minimal de dependencias

- Paso 2: ...
  - $CE \rightarrow A$, $CE \rightarrow F$,
    - $C^+=\{C, A\}$, E no pertenece a C^+ luego E no es raro con respecto a C
    - $E^+=\{E\}$, C no pertenece a E^+ luego C no es raro con respecto a E

  luego $CE \rightarrow A$, $CE \rightarrow F$ se quedan como están.

$F^{(1)}=\{AB \rightarrow C, D \rightarrow E, D \rightarrow F, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CF \rightarrow B, CF \rightarrow D, ACD \rightarrow B, CE \rightarrow A, CE \rightarrow F\}$

Recubrimiento minimal de dependencias

- El resultado de este paso es:

  $F^{(2)}=\{AB\rightarrow C,\ D\rightarrow E,\ D\rightarrow F,\ C\rightarrow A,\ BE\rightarrow C,\ BC\rightarrow D,\ CF\rightarrow B,\ CF\rightarrow D,\ CD\rightarrow B,\ CE\rightarrow A,\ CE\rightarrow F\}$

Recubrimiento minimal de dependencias

- Paso 3: obtención de $F^{(3)}$ o F' mediante eliminación de dependencias redundantes.
- Una dependencia $\alpha \rightarrow \beta \in F$ es redundante si se puede obtener a partir de las demás mediante aplicación de los axiomas y las reglas de Armstrong en una secuencia finita de pasos, es decir, sii:

$$\alpha \rightarrow \beta \in (F - \{\alpha \rightarrow \beta\})^+$$

- Difícil de comprobar

Recubrimiento minimal de dependencias

- Paso 3: ...
  - Pero existe una relación entre el cierre de dependencias y el cierre de atributos, y éste último es más fácil de comprobar:

$$\alpha \rightarrow \beta \in (F-\{\alpha \rightarrow \beta\})^{+} \Leftrightarrow \beta \in \alpha^{+}_{F-\{\alpha \rightarrow \beta\}}$$

  - Cada dependencia redundante se suprime.

Recubrimiento minimal de dependencias

- Paso 3:
  - $AB \rightarrow C$ es redundante si $C \in \{AB\}^{+}_{F^{(2)}-\{AB \rightarrow C\}}$
    - $\{AB\}^{+}_{F^{(2)}-\{AB \rightarrow C\}} = \{A, B\}$, $C \notin \{AB\}^{+}_{F^{(2)}-\{AB \rightarrow C\}}$ luego $AB \rightarrow C$ no es redundante y aparece en $F^{(3)}$
  - $D \rightarrow E$ es redundante si $E \in D^{+}_{F^{(2)}-\{D \rightarrow E\}}$
    - $D^{+}_{F^{(2)}-\{D \rightarrow E\}} = \{D, F\}$, $E \notin D^{+}_{F^{(2)}-\{D \rightarrow E\}}$ luego $D \rightarrow E$ no es redundante y aparece en $F^{(3)}$

$F^{(2)} = \{AB \rightarrow C, D \rightarrow E, D \rightarrow F, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CF \rightarrow B, CF \rightarrow D, CD \rightarrow B, CE \rightarrow A, CE \rightarrow F\}$

Recubrimiento minimal de dependencias

- Paso 3:
  - $D \rightarrow F$ es redundante si $F \in D^{+}_{F^{(2)}-\{D\rightarrow F\}}$
    - $D^{+}_{F^{(2)}-\{D\rightarrow F\}}=\{D, E\}$, $F \notin D^{+}_{F^{(2)}-\{D\rightarrow F\}}$ luego $D \rightarrow F$ no es redundante y aparece en $F^{(3)}$
  - $C \rightarrow A$ es redundante si $A \in C^{+}_{F^{(2)}-\{C\rightarrow A\}}$
    - $C^{+}_{F^{(2)}-\{C\rightarrow A\}}=\{C\}$, $A \notin C^{+}_{F^{(2)}-\{C\rightarrow A\}}$ luego $C \rightarrow A$ no es redundante y aparece en $F^{(3)}$

$F^{(2)}=\{AB\rightarrow C,\ D\rightarrow E,\ D\rightarrow F,\ C\rightarrow A,\ BE\rightarrow C,\ BC\rightarrow D,\ CF\rightarrow B,\ CF\rightarrow D,\ CD\rightarrow B,\ CE\rightarrow A,\ CE\rightarrow F\}$

Recubrimiento minimal de dependencias

- Paso 3:
  - $BE \rightarrow C$ es redundante si $C \in \{BE\}^{+}_{F^{(2)}-\{BE \rightarrow C\}}$
    - $\{BE\}^{+}_{F^{(2)}-\{BE \rightarrow C\}} = \{B, E\}$, $C \notin \{BE\}^{+}_{F^{(2)}-\{BE \rightarrow C\}}$ luego $BE \rightarrow C$ no es redundante y aparece en $F^{(3)}$
  - $BC \rightarrow D$ es redundante si $D \in \{BC\}^{+}_{F^{(2)}-\{BC \rightarrow D\}}$
    - $\{BC\}^{+}_{F^{(2)}-\{BC \rightarrow D\}} = \{B, C, A\}$, $D \notin \{BC\}^{+}_{F^{(2)}-\{BC \rightarrow D\}}$ luego $BC \rightarrow D$ no es redundante y aparece en $F^{(3)}$

$F^{(2)} = \{AB \rightarrow C, D \rightarrow E, D \rightarrow F, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CF \rightarrow B, CF \rightarrow D, CD \rightarrow B, CE \rightarrow A, CE \rightarrow F\}$

Recubrimiento minimal de dependencias

- Paso 3:
  - $CF \rightarrow B$ es redundante si $B \in \{CF\}^+_{F^{(2)}-\{CF \rightarrow B\}}$
    - $\{CF\}^+_{F^{(2)}-\{CF \rightarrow B\}} = \{C, F, A, D, B, E\}$, $B \in \{CF\}^+_{F^{(2)}-\{CF \rightarrow B\}}$ luego $CF \rightarrow B$ **es redundante** y no aparece en $F^{(3)}$
  - $CF \rightarrow D$ es redundante si $D \in \{CF\}^+_{F^{(2)}-\{CF \rightarrow D\}}$
    - $\{CF\}^+_{F^{(2)}-\{CF \rightarrow D\}} = \{C, F, A\}$, $D \notin \{CF\}^+_{F^{(2)}-\{CF \rightarrow D\}}$ luego $CF \rightarrow D$ no es redundante y aparece en $F^{(3)}$

$F^{(2)} = \{AB \rightarrow C, D \rightarrow E, D \rightarrow F, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CF \rightarrow B, CF \rightarrow D, CD \rightarrow B, CE \rightarrow A, CE \rightarrow F\}$

Recubrimiento minimal de dependencias

- Paso 3:
  - $CD \rightarrow B$ es redundante si $B \in \{CD\}^{+}_{F^{(2)}-\{CD \rightarrow B\}}$
    - $\{CD\}^{+}_{F^{(2)}-\{CD \rightarrow B\}} = \{C, D, E, F, A\}$, $B \notin \{CD\}^{+}_{F^{(2)}-\{CD \rightarrow B\}}$ luego $CD \rightarrow B$ no es redundante y aparece en $F^{(3)}$
  - $CE \rightarrow A$ es redundante si $A \in \{CE\}^{+}_{F^{(2)}-\{CE \rightarrow A\}}$
    - $\{CE\}^{+}_{F^{(2)}-\{CE \rightarrow A\}} = \{C, E, A, F, D, B\}$, $A \in \{CE\}^{+}_{F^{(2)}-\{CE \rightarrow A\}}$ luego $CE \rightarrow A$ **es redundante** y no aparece en $F^{(3)}$

$F^{(2)} = \{AB \rightarrow C, D \rightarrow E, D \rightarrow F, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CF \rightarrow B, CF \rightarrow D, CD \rightarrow B, CE \rightarrow A, CE \rightarrow F\}$

Ignacio J. Blanco, Jesús Campaña, Carlos Cruz, María Ros, Daniel Sánchez

Recubrimiento minimal de dependencias

- Paso 3:
  - $CE \rightarrow F$ es redundante si $F \in \{CE\}^+_{F^{(2)}-\{CE \rightarrow F\}}$
    - $\{CE\}^+_{F^{(2)}-\{CE \rightarrow F\}} = \{C, E, A\}$, $F \notin \{CE\}^+_{F^{(2)}-\{CE \rightarrow F\}}$ luego $CE \rightarrow F$ no es redundante y aparece en $F^{(3)}$

$F^{(2)} = \{AB \rightarrow C, D \rightarrow E, D \rightarrow F, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CF \rightarrow B, CF \rightarrow D, CD \rightarrow B, CE \rightarrow A, CE \rightarrow F\}$

El resultado es:

$$F^{(3)} = \{AB \rightarrow C, D \rightarrow E, D \rightarrow F, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CF \rightarrow D, CD \rightarrow B, CE \rightarrow F\} = F'$$

Ignacio J. Blanco, Jesús Campaña, Carlos Cruz, María Ros, Daniel Sánchez

Recubrimiento minimal de dependencias

- Dado que tanto *F* como *F'* son equivalentes (tienen cierres iguales), siempre es mejor trabajar con F' puesto que las dependencias han sido simplificadas, lo cual es más útil a la hora de realizar procesos posteriores como el cálculo de llaves candidatas (si no se conocieran) o la normalización.

Recubrimiento minimal de dependencias

- Otra de las ventajas de trabajar con F' es que, dado que no tiene dependencias redundantes, cuando una dependencia no aparece en ninguno de los dos conjuntos resultantes de la aplicación del Teorema de Heath, significa que esa dependencia se ha perdido y que la descomposición no preserva dependencias.

Cálculo de llaves candidatas

Dado el esquema R con un conjunto de dependencias funcionales F, el procedimiento para el cálculo de las llaves candidatas es el siguiente:

1. Eliminación de atributos independientes:

   Construir, a partir de R, un conjunto de atributos R_{si} en el que se han eliminado los atributos independientes, dado que estos participan en cualquier clave candidata y de ellos no se puede deducir ningún otro (salvo ellos mismos).

Cálculo de llaves candidatas

2. Eliminación de atributos equivalentes:

- Construir, a partir de R_{si}, un conjunto de atributos R_{sie} en el que se han eliminado los atributos equivalentes, escogiendo uno de los dos atributos de cada equivalencia y sustituyendo el eliminado por el elegido en cada dependencia funcional de F en la que aparezca;
- Como resultado de este paso, puede darse el caso de que determinados atributos aparezcan como independientes entre sí.

Cálculo de llaves candidatas

3. Selección de una clave de Rsie en la que no aparecen determinantes que sean determinados:
   - Se selecciona como primer candidato a clave candidata Kp cualquier determinante de Rsie que no sea determinado;
     - Si no quedan más determinantes en Rsie que sean a la vez determinados, Kp es clave candidata y se pasa al paso 5. En caso contrario, se pasa al paso 4.

Ignacio J. Blanco, Jesús Campaña, Carlos Cruz, María Ros, Daniel Sánchez

Cálculo de llaves candidatas

4. Selección de una clave de R_{sie} en el que pueden aparecer determinantes que puedan ser determinados:

a) Se construye el conjunto R'_{sie} eliminando de R_{sie} aquellos atributos que aparecen en K_p+ y no están implicados en otras dependencias que no sean las necesarias para calcular K_p+.

   I. Se obtiene una clave provisional K'_p en R'sie, con los determinantes de K_p y añadiendo a estos un nuevo determinante que sea determinado. Si $K'_p+ = R'_{sie}$, entonces K'_p es una clave de R'_{sie}. En caso contrario, se añade un nuevo atributo que sea determinado y que no pertenezca al cierre de K'_p y se vuelve a comprobar.

   II. Se repite la operación para cubrir todas las claves posibles.

   III. Se añade a cada clave de R'sie las obtenidas del paso 3 para obtener las claves de Rsie.

b) Si no se pudiese construir R'_{sie}, se procede considerando R_{sie} como R'_{sie}.

Cálculo de llaves candidatas

5. Añadir los atributos independientes a las claves obtenidas para Rsie.
6. Replicar las claves con las equivalencias eliminadas en el paso 2 para generar todas las claves.

Ignacio J. Blanco, Jesús Campaña, Carlos Cruz, María Ros, Daniel Sánchez

Cálculo de llaves candidatas

- Ejemplo 1

  Sea el esquema de relación:

  R({A,B,C,D,E,F,G,H,I,J}; {AB→C, C→AB, E→D, D→E, E→F, F→E, ABD →G, CF→H})

- Paso 1
  - Los atributos I y J son independientes porque no forman parte de ninguna DF, luego, en este primer paso se eliminan de la relación:
  - Rsi ({A,B,C,D,E,F,G,H}, {AB↔C, D↔E↔F, ABD→G, CF→H})
- Paso 2
  - Existen dos grupos de descriptores equivalentes:
    a) AB y C
    b) D, E y F
  - Del grupo a) nos quedaríamos, por ejemplo, con C y del grupo b) con D (eliminaríamos, por tanto, AB, E y F); la relación resultante sin equivalencias sería:
  - Rsie ({C, D, G, H};{CD→G, CD→H})

Cálculo de llaves candidatas

- Paso 3
  - En la Rsie anterior, CD es el único implicante, pero no implicado, luego una Kp sería CD, como el resto son sólo implicados, CD es clave de Rsie (no haría falta hallar el cierre de CD). Pasaríamos al paso 5.
- Paso 5
  - Si a CD le añadimos los atributos independientes I y J tenemos CDIJ que es la clave de R.
- Paso 6
  - Los descriptores equivalentes eran: AB↔C y D↔E↔F
  - La clave CDIJ genera las siguientes claves candidatas de R:
    {C|AB}{D|E|F}IJ
  - En total, son 6 claves: CDIJ, CEIJ, CFIJ, ABDIJ, ABEIJ y ABFIJ

Cálculo de llaves candidatas

Ejemplo 2

Sea una relación de esquema R = {A, B, C, D, E} que verifica el conjunto de dependencias funcionales:

F= {ADE -> B, AC -> D, A -> B, DE -> C, D -> AB}

Calcular todas las claves candidatas existentes en la relación R y justifica por qué son las únicas existentes

Cálculo de llaves candidatas

Agrupar los grupos de atributos:

-Atributos independientes (no aparecen en ninguna dependencia): Ø
-Atributos que aparecen a la izquierda en todas las dependencias en las que aparecen: {E}
-Atributos que aparecen a la izquierda de alguna dependencia y a la derecha de otra: {A, C, D}
-Atributos que aparecen a la derecha en todas las dependencias en las que aparecen: {B}

R = {A, B, C, D, E} y F= {ADE -> B, AC -> D, A -> B, DE -> C, D -> AB}

Cálculo de llaves candidatas

Paso 1:
Eliminar atributos independientes:

No hay ninguno, por tanto Rsi = R

Paso 2:
Eliminar atributos equivalentes:

No hay equivalencias entre parejas de atributos;
por tanto Rsie = Rsi = R y Fsie = F

Rsie = {A, B, C, D, E} y Fsie= {ADE -> B, AC -> D, A -> B, DE -> C, D -> AB}

Cálculo de llaves candidatas

Paso 3:
Procesar atributos que aparecen solo a la izquierda:

Kp = {E} donde Kp es conjunto de claves candidatas

- Hay un solo candidato, probar si es clave (cierre de candidato = Rsie):

  $E^+ = \{E\} \neq$ Rsie, por tanto no es clave y CKsie = Ø

- Al no haber clave (CKsie = Ø) en este paso entonces pasamos al paso 4

Rsie = {A, B, C, D, E} y Fsie= {ADE -> B, AC -> D, A -> B, DE -> C, D -> AB}

Cálculo de llaves candidatas

Paso 4:
Procesar atributos que aparecen a la izquierda y a la derecha:

- Combinar el candidato anterior con cada atributo que esté a la izquierda y la derecha ({A, C, D}), y que no aparezca en su cierre (E^+ = {E})

Kp’ = {EA, EC, ED}

- Probar si son claves

EA^+ = {E, A, B} ≠ Rsie, por tanto no es clave y

CKsie = Ø, Kp’ = {EC, ED}

- Como EA no es clave hay que combinar este candidato con cada atributo que esté a la izquierda y la derecha ({A, C, D}), y que no aparezca en su cierre (EA^+ = {E, A, B})

EAC, es extensión de candidato pendiente EC; no se incluye en Kp’

EAD, es extensión de candidato pendiente ED; no se incluye en Kp’

R = {A, B, C, D, E} y F= {ADE -> B, AC -> D, A -> B, DE -> C, D -> AB}

Cálculo de llaves candidatas

Paso 4:
Procesar atributos que aparecen a la izquierda y a la derecha:

Kp' = {EC, ED}

- Probar si son claves

EC^+ = {E, C} ≠ Rsie, por tanto no es clave y

CKsie = Ø, Kp' = {ED}

- Como EC no es clave hay que combinar este candidato con cada atributo que esté a la izquierda y la derecha ({A, C, D}), y que no aparezca en su cierre (EC^+ = {E, C})

ECA, no es extensión de candidato pendiente, ni de ninguna clave (CKsie = Ø) ; se incluye en Kp' = {ED, ECA}

ECD, es extensión de candidato pendiente ED; no se incluye en Kp'

R = {A, B, C, D, E} y F= {ADE -> B, AC -> D, A -> B, DE -> C, D -> AB}

Cálculo de llaves candidatas

Paso 4:
Procesar atributos que aparecen a la izquierda y a la derecha:

Kp’ = Kp’ = {ED, ECA}

- Probar si son claves

ED$^+$ = {E, D, C, A, B} = Rsie, por tanto es clave y

CKsie = ED, Kp’ = {ECA}

ECA$^+$ = {E, C, A, D, B} = Rsie, por tanto es clave y

CKsie = {ED,ECA} y Kp’ = {Ø}

Como no hay mas candidatos, termina el paso 4

R = {A, B, C, D, E} y F= {ADE -> B, AC -> D, A -> B, DE -> C, D -> AB}

Cálculo de llaves candidatas

Paso 5:
Incorporar atributos independientes:
- Como no hay, CK’ = CKsie = {ED, ECA}

Paso 6:
Incorporar atributos equivalentes:
- Como no hay, se mantiene CK= CK’ = CKsie = {ED, ECA}

R = {A, B, C, D, E} y F= {ADE -> B, AC -> D, A -> B, DE -> C, D -> AB}

Dependencia funcional transitiva

Sea R un esquema de relación, F un conjunto de DFs asociado y $CK \subset R$ una clave candidata de R. Decimos que $CK \rightarrow \beta$, con β formado por algún atributo no primo, es transitiva si $\exists \alpha \subset R \mid \alpha \rightarrow R \notin F^{+}$ tal que

$$CK \rightarrow \alpha \in F \quad y \quad \alpha \rightarrow \beta \in F$$

Es decir, la dependencia $CK \rightarrow \beta$ que debe cumplirse por ser CK clave candidata se verifica a través de $CK \rightarrow \alpha$, $\alpha \rightarrow \beta$ y el axioma de transitividad de Armstrong.

3ra. Forma Normal – 3FN

Decimos que una relación R está en tercera forma normal (3FN) sii

- Está en segunda forma normal.
- No presenta dependencias transitivas problemáticas.

3ra. Forma Normal – 3FN

Ejemplo

La relación ASIGNATURA (#asig, nombre, curso, plan, ct, cp, coste), con #asig como clave primaria, presenta una dependencia funcional transitiva:

- #asig $\rightarrow$ nombre curso plan ct cp
- plan ct cp $\rightarrow$ coste

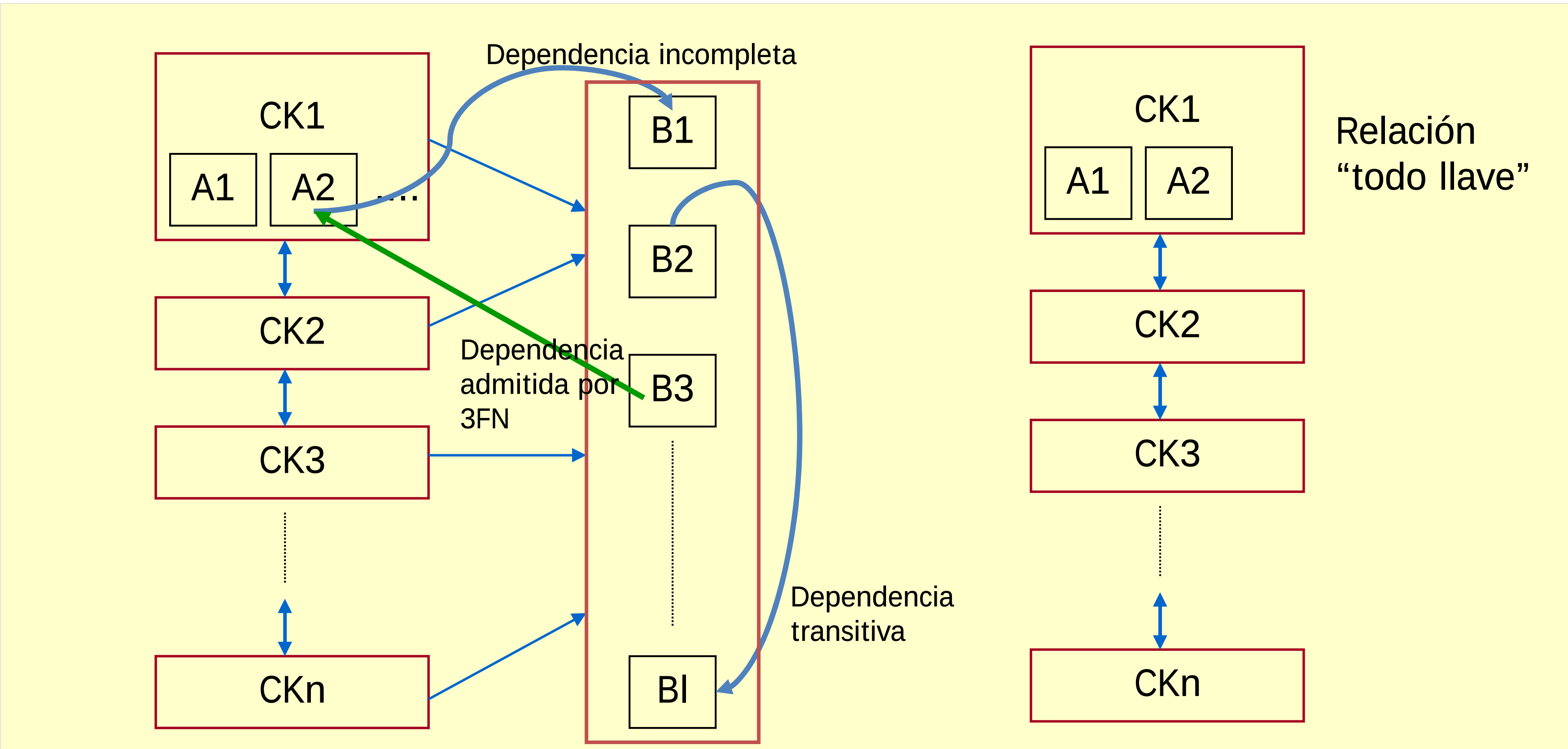
que es el origen de su "mal comportamiento".

Normalización: Descomposición sin pérdidas

- R_1(#asig, nombre, curso, plan, ct, cp)
  - PK: #asig, dependencia "directa" (no transitiva)
- R_2(plan, ct, cp, coste)
  - PK: (plan, ct, cp), dependencia "directa" (no transitiva)

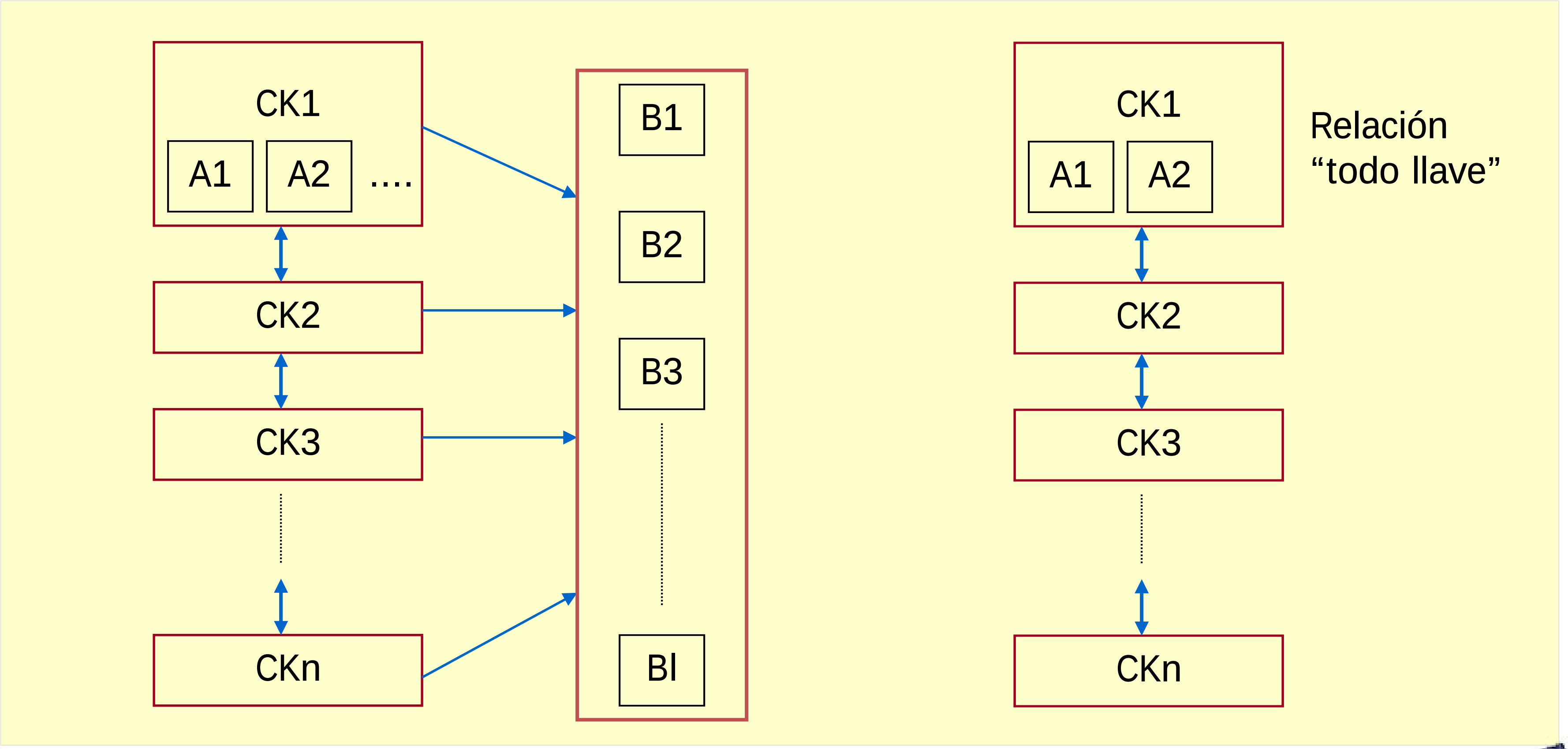
Dependencia funcional transitiva

Posibles estructuras de DFs



Dependencia funcional transitiva

Estructuras "correctas" de dependencias funcionales



Ignacio J. Blanco, Jesús Campaña, Carlos Cruz, María Ros, Daniel Sánchez

3ra. Forma Normal – 3FN

- La definición original de 3FN dada por Codd tiene deficiencias ya que no produce diseños satisfactorios cuando:
  - Hay varias claves candidatas
  - Las claves candidatas son compuestas
  - Las claves candidatas se solapan.

Forma Normal Boyce-Codd– FNBC

- La FNBC considera estos casos:
  más restrictiva que la 3FN, aunque equivalente a ésta si no se dan las anteriores condiciones.
- Definición formal:

  Dada una relación R y F su conjunto de DFs asociado, decimos que R está en forma normal de Boyce y Codd (FNBC) si y sólo si $\forall \alpha \rightarrow \beta \in F$ se verifica:

  - α es llave candidata y $\beta \not\subset \alpha$

  o bien

  - $\beta \subseteq \alpha$

Forma Normal Boyce-Codd– FNBC

- Definición:
  - **Determinante de una relación**: Todo conjunto de atributos del cual depende de forma completa otro atributo de la relación.
- FNBC, definición alternativa:
  - Dada una relación R y F su conjunto de DFs asociado, decimos que R está en forma normal de Boyce y Codd (FNBC) si y sólo si todo determinante es una clave candidata.

Ejercicio Normalización

Sea **R ={A, B, C, D, E, F}** una relación de esquema, con un conjunto de claves candidatas **CK={DE, BE}** y que verifica un conjunto F de dependencias funcionales

F = {D →C, DE → F, B → D, AF → C, DF → A, D → B}

Comprobar si la relación está en Forma Normal de Boyce y Codd, y si no lo está, realizar una descomposición hasta que todas las relaciones de la descomposición estén en dicha forma normal.

Ejercicio Normalización

R={A, B, C, D, E, F} , CK={DE, BE}, F={D→C, DE→ F, B→ D, AF→ C, DF→ A, D→ B}

R no está en Forma Normal de Boyce y Codd (BCNF) porque:

D → C, B → D, AF → C, DF → A y D → B son dependencias de F, y {D, B, AF y DF} son determinantes (están a la izquierda) y no son claves candidatas de CK

Por tanto,
Aplicar Teorema de Heath sobre dependencias funcionales de F que no cumplan la forma normal

Ejercicio Normalización

R={A, B, C, D, E, F} , CK={DE, BE}, F={D→C, DE→ F, B→ D, AF→ C, DF→ A, D→ B}

Por tanto,
Aplicar Teorema de Heath sobre dependencias funcionales de F que no cumplan la forma normal.

¿Cuál elegir?
Ver qué atributo a la derecha puede causar menos daño al aplicarle Heath.

Para ello:

- Dependencias cuyo atributo a la derecha sólo está a la derecha en F.
- Dependencias cuyo atributo a la derecha está a la izquierda de otra dependencia en F.
  - Y que el atributo a la derecha no pertenece a ninguna clave candidata
  - Y que el atributo a la derecha pertenezca a alguna clave candidata

Ejercicio Normalización

R={A, B, C, D, E, F} , CK={DE, BE}, F={D→C, DE→ F, B→ D, AF→ C, DF→ A, D→ B}

Para ello, escogemos:

- Dependencias cuyo atributo a la derecha sólo está a la derecha en F

  {C}: {D→C, AF→ C}

- Dependencias cuyo atributo a la derecha está a la izquierda de otra dependencia en F
  - Y que el atributo a la derecha no pertenece a ninguna clave candidata

  {A}: {DF→ A}

  - Y que el atributo a la derecha pertenezca a alguna clave candidata

  {B, D}: {B→ D, D→ B}

Ejercicio Normalización

R={A, B, C, D, E, F} , CK={DE, BE}, F={D→C, DE→ F, B→ D, AF→ C, DF→ A, D→ B}

Para ello, escogemos:

- Dependencias cuyo atributo a la derecha sólo está a la derecha en F

  {C}: {D→C, AF→ C}

Aplicamos Heath a una de las dos dependencias de C:

R1={C, D}, F1={D→C}, CK1={D}

R2={A, B, D, E, F}, F2={DE→ F, B→ D, DF→ A, D→ B}

CK2=CK={DE, BE}

Ejercicio Normalización

R={A, B, C, D, E, F} , CK={DE, BE}, F={D→C, DE→ F, B→ D, AF→ C, DF→ A, D→ B}

Aplicamos Heath a una de las dos dependencias de C:

R1={C, D}, F1={D→C}, CK1={D}, **R1 está en FNBC** (todas sus dependencias tienen una CK a la izquierda)

R2={A, B, D, E, F}, F2={DE→ F, B→ D, DF→ A, D→ B}

CK2=CK={DE, BE}

R2 no está en FNBC, culpa de {B→ D, DF→ A, D→ B}

Por tanto, hay que aplicar Heath a R2 (se repite el algoritmo)

Ejercicio Normalización

R={A, B, C, D, E, F} , CK={DE, BE}, F={D→C, DE→ F, B→ D, AF→ C, DF→ A, D→ B}

R2={A, B, D, E, F}, F2={DE→ F, B→ D, DF→ A, D→ B}
CK2=CK={DE, BE}

- Dependencias cuyo atributo a la derecha sólo está a la derecha en F2
  {A}: {DF→ A}

R21= {A,D,F}, F21= {DF→ A}, CK21= {DF}, **R21 está en FNBC** (todas sus dependencias tienen una CK a la izquierda)

R22={B,D,E,F}, F22= {DE→ F, B→ D, D→ B}, CK22=CK2=CK={DE, BE}
R22 no está en FNBC, culpa de {B→ D, D→ B},
Por tanto, aplicarle el algoritmo

Ejercicio Normalización

R={A, B, C, D, E, F} , CK={DE, BE}, F={D→C, DE→ F, B→ D, AF→ C, DF→ A, D→ B}

R2={A, B, D, E, F}, F2={DE→ F, B→ D, DF→ A, D→ B}
CK2=CK={DE, BE}

R22={B,D,E,F}, F22= {DE→ F, B→ D, D→ B}, CK22=CK2=CK={DE, BE}

- Dependencias cuyo atributo a la derecha sólo está a la derecha en F22
  {DE→ F}, esta no es problemática

- Dependencias cuyo atributo a la derecha está a la izquierda de otra dependencia en F22
  - Y que el atributo a la derecha no pertenece a ninguna clave candidata
  {B→ D}

  - Y que el atributo a la derecha pertenezca a alguna clave candidata

Ejercicio Normalización

R={A, B, C, D, E, F} , CK={DE, BE}, F={D→C, DE→ F, B→ D, AF→ C, DF→ A, D→ B}

R2={A, B, D, E, F}, F2={DE→ F, B→ D, DF→ A, D→ B}

CK2=CK={DE, BE}

R22={B,D,E,F}, F22= {DE→ F, B→ D, D→ B}, CK22=CK2=CK={DE, BE}

- Dependencias cuyo atributo a la derecha está a la izquierda de otra dependencia en F22
  - Y que el atributo a la derecha no pertenece a ninguna clave candidata: {B→ D}

R221={B,D}, F221= {B→ D, D→ B}, CK221={B,D} **R221 está en FNBC** (todas sus dependencias tienen una CK a la izquierda

R222={D,E,F}, }, F22= {DE→ F}, CK222={DE} **R222 está en FNBC** (todas sus dependencias tienen una CK a la izquierda

Ejercicio Normalización

R={A, B, C, D, E, F} , CK={DE, BE}, F={D→C, DE→ F, B→ D, AF→ C, DF→ A, D→ B}

Tablas resultantes:

R1={C, D}, F1={D→C}, CK1={D}

R21= {A,D,F}, F21= {DF→ A}, CK21= {DF},

R221={B,D}, F221= {B→ D, D→ B}, CK221={B,D}

R222={D,E,F}, }, F22= {DE→ F}, CK222={DE}

{ ({C, D}, r1), ({A,D,F}, r21), ({B,D},r221), ({D,E,F}, r222) }

Ejercicio Normalización

R={A, B, C, D, E, F} , CK={DE, BE}, F={D→C, DE→ F, B→ D, AF→ C, DF→ A, D→ B}

Tablas resultantes:

R1={C, D}, F1={D→C}, CK1={D}

R21= {A,D,F}, F21= {DF→ A}, CK21= {DF},

R221={B,D}, F221= {B→ D, D→ B}, CK221={B,D}

R222={D,E,F}, }, F22= {DE→ F}, CK222={DE}

¿Se pierde alguna dependencia funcional de F?

Ejercicio Normalización

R={A, B, C, D, E, F} , CK={DE, BE}, F={D→C, DE→ F, B→ D, AF→ C, DF→ A, D→ B}

R1={C, D}, F1={D→C}, CK1={D}
R21= {A,D,F}, F21= {DF→ A}, CK21= {DF},
R221={B,D}, F221= {B→ D, D→ B}, CK221={B,D}
R222={D,E,F}, F22= {DE→ F}, CK222={DE}

¿Se pierde alguna dependencia funcional de F?

¿AF→ C?

Basta comprobar si C está en cierre de AF^+

AF^+ ={A,F}, C no está, por tanto se pierde esta dependencia funcional