Normalización de Esquemas

Base de Datos

Mónica Caniupán
mcaniupan@ubiobio.cl

Universidad del Bío-Bío

2020

¿Por Qué Refinar Esquemas?

- Uno de los problemas en BDs es el almacenamiento redundante de la información:
  1. Almacenamiento redundante: alguna información se almacena de manera repetida
  2. Anomalías de actualización: se generan inconsistencias si algunas copias no se mantienen actualizadas
  3. Anomalías de inserción: puede que no sea posible almacenar determinada información a menos que otra información, sin relación alguna con ella, también se almacene
  4. Anomalía de eliminación: podría no ser posible eliminar cierta información sin perder otra

Ejemplo: Anomalías

- Supongamos la siguiente relación *Horas_Emp*:

Horas_Emp					
id	nombre	depto	categoria	horas_extras	horas_trab
11	*juan*	48	8	10	40
12	*pedro*	22	8	10	30
13	*enrique*	35	5	7	30
14	*maria*	35	6	8	32
15	*carolina*	35	8	10	40

- Supongamos además que el atributo *horas_extras* depende del atributo *categoria*, i.e., por cada valor de categoría existe un único valor para *horas_extras*
- Almacenamiento redundante: para la categoría 8 corresponden 10 horas extras, lo cual está almacenado tres veces

Ejemplo: Anomalías

Horas_Emp					
id	nombre	depto	categoria	horas_extras	horas_trab
11	*juan*	48	8	10	40
12	*pedro*	22	8	10	30
13	*enrique*	35	5	7	30
14	*maria*	35	6	8	32
15	*carolina*	35	8	10	40

- Anomalías de actualización: si se modifica el valor de *horas_extras* en la primera tupla y no se modifica en el resto de las tuplas con categoría 8 se producen inconsistencias
- Anomalías de inserción: no podemos insertar un nuevo empleado a menos que se conozca el número de horas extras para la categoría del empleado
- Anomalías de eliminación: si borramos todas las tuplas para una categoría determinada, perdemos la información de la categoría y sus horas extras (ejemplo, si borramos el empleado con id=14)

Origen de Redundancia en Esquemas de BDs

- La redundancia surge cuando un esquema relacional fuerza una asociación entre atributos que no es natural
  - Ejemplo: en la relación *Horas_Emp* el atributo *horas_extras* no tiene relación directa con el atributo *depto*
- Se pueden utilizar las dependencias funcionales (DFs) para identificar estas situaciones y sugerir maneras de refinar (cambiar) los esquemas
- La idea fundamental es que muchos problemas que surgen de la redundancia se pueden abordar sustituyendo una relación dada por un conjunto de relaciones “mas pequeñas”

Descomposición de Relaciones

- La descomposición del esquema de una relación *R* consiste en sustituir el esquema de la relación por dos (o más) esquemas que contienen un subconjunto de los atributos de *R* y, conjuntamente, incluyen todos los atributos de *R*
- Ejemplo: La relación *Horas_Emp* se puede descomponer en dos relaciones:
  - Horas_Emp(id, nombre, depto, categoria, horas_trab)
  - Categoria_Horas(categoria, horas_extras)

Horas_Emp				
id	nombre	depto	categoria	horas_trab
11	*juan*	48	8	40
12	*pedro*	22	8	30
13	*enrique*	35	5	30
14	*maria*	35	6	32
15	*carolina*	35	8	40

Categoria_Horas	
categoria	horas_extras
8	10
5	7
6	8

Problemas con la Descomposición

- La descomposición del esquema de una relación puede crear más problemas de los que soluciona
- Se deben formular repetidamente dos preguntas importantes:
  1. ¿Hace falta descomponer la relación?
  2. ¿Qué problemas provoca una descomposición dada?

Problemas con la Descomposición

- Para ayudar con la primer pregunta ¿Hace falta descomponer la relación? se han propuesto varias formas normales (FNs) para las relaciones
- Si el esquema de una relación está en alguna FN, se sabe que no pueden surgir determinadas clases de problemas
- Con respecto a la segunda pregunta ¿Qué problemas provoca una descomposición dada? resultan de especial interés dos propiedades de las descomposiciones:
  1. Reunión sin pérdida: la descomposición permite recuperar cualquier instancia de la relación descompuesta a partir de las instancias de las relaciones de menor tamaño
  2. Conservación de las dependencias: la descomposición permite hacer que se cumpla cualquier restricción de la relación original con sólo hacer que se cumplan algunas restricciones en cada una de las relaciones de menor tamaño

Dependencias Funcionales

- Una dependencia funcional (DF) es una restricción de integridad que generaliza el concepto de clave
- Sea R el esquema de una relación y sean X, Y conjuntos no vacíos de atributos de R. Se dice que una instancia r de R satisface la DF: $X \rightarrow Y$ si para cada par de tuplas t_1 y t_2 en r se cumple:

$$\text{Si } t_1.X = t_2.X, \text{ entonces } t_1.Y = t_2.Y$$

  donde $t_1.X$ es la proyección de los atributos X en la tupla t_1
- $X \rightarrow Y$ se lee X determina (funcionalmente) a Y
- En otras palabras, $X \rightarrow Y$ indica básicamente que si dos tuplas de r coinciden en el valor de los atributos de X, también deben coincidir en el valor de los atributos de Y

Dependencias Funcionales

- Para nuestro ejemplo:
  - R es $Horas\_Emp(id, nombre, depto, categoria, horas\_extras)$
  - r es la instancia del ejemplo:

Horas_Emp					
id	nombre	depto	categoria	horas_extras	horas_trab
11	*juan*	48	8	10	40
12	*pedro*	22	8	10	30
13	*enrique*	35	5	7	30
14	*maria*	35	6	8	32
15	*carolina*	35	8	10	40

  - X es *categoria* e Y es *horas_extras* y la DF es $categoria \rightarrow horas\_extras$
  - Es fácil mostrar que para todo par de tuplas t_1 y t_2 en r se cumple:

Si $t_1.categoria = t_2.categoria$, entonces $t_1.horas\_extras = t_2.horas\_extras$

Ejemplo: Dependencias Funcionales

- La siguiente relación satisface la DF $AB \rightarrow C$:

A	B	C	D
a_1	b_1	c_1	d_1
a_1	b_1	c_1	d_2
a_2	b_1	c_3	d_1

- Si se inserta la tupla $\langle a_1, b_1, c_2, d_1 \rangle$ a r, entonces r no satisface la DF
- Las dos primeras tuplas muestran que las DF no son lo mismo que las restricciones de claves: es evidente que AB no es clave de la relación ¿Por qué?

Instancias Legales y Claves Primarias

- Una instancia de una relación R es legal si satisface todas las restricciones de integridad sobre R, incluyendo las DFs
- Examinando las instancias de una relación se puede saber si una DF no se cumple
- Las restricciones de clave primaria son un caso especial de DFs
  - Los atributos en la llave desempeñan el papel de X y el conjunto de todos los atributos de la relación interpretan el de Y
- La definición de DFs no exige que el conjunto X sea mínimo, sin embargo para que X sea clave éste debe ser mínimo

Ejemplo: Clave Primaria

- En el esquema *Navegantes*(**idn**, *nombre*, *categoria*, *edad*), la DF que define la clave primaria es:

$$idn \rightarrow idn, nombre, categoria, edad$$

- Por cada valor de *idn* existe un valor único para (*idn*), *nombre, categoria* y *edad*
  - En $idn \rightarrow idn, nombre, categoria, edad$, el conjunto *idn* es minimal
  - En $idn, nombre \rightarrow categoria, edad$, el conjunto *idn*, *nombre* no es minimal, ya que sabemos que $idn \rightarrow idn, nombre, categoria, edad$

Razonamiento sobre las DFs

- Dado un conjunto de DFs sobre el esquema de una relación R, normalmente se cumplen varias DFs adicionales sobre R siempre que se cumplan todas las DFs dadas
- Consideremos el esquema:
  $\mathit{Trabajadores}(\underline{\mathit{idt}}, \mathit{nombre}, \mathit{plaza}, \mathit{depto}, \mathit{desde})$
- Dado que *idt* es clave, se cumple: $\mathit{idt} \rightarrow \mathit{nombre}, \mathit{plaza}, \mathit{depto}, \mathit{desde}$, consideremos además que se cumple la DF: $\mathit{depto} \rightarrow \mathit{plaza}$
  - En cualquier instancia de *Trabajadores* si dos tuplas tienen el mismo valor de *idt*, deben tener el mismo valor de *depto*, ya que $\mathit{idt} \rightarrow \mathit{depto}$
  - Por la segunda DF $\mathit{depto} \rightarrow \mathit{plaza}$ también deben tener el mismo valor de *plaza*
  - Por lo tanto, también se cumple la DF $\mathit{idt} \rightarrow \mathit{plaza}$

Clausura de un conjunto de DFs

- El conjunto de todas las DFs implícitas en un conjunto dado de DFs D se denomina clausura de D (cierre) y se denota por D^+
- Para obtener D^+ se pueden aplicar reiteradamente los axiomas de Armstrong
- Sean X, Y y Z conjuntos de atributos de un esquema R:
  - Reflexividad: Si $Y \subseteq X$, entonces $X \rightarrow Y$
  - Aumento: Si $X \rightarrow Y$, entonces $XZ \rightarrow YZ$ para cualquier Z
  - Transitividad: Si $X \rightarrow Y$ y $Y \rightarrow Z$, entonces $X \rightarrow Z$
- Reglas Adicionales:
  - Unión: Si $X \rightarrow Y$ y $X \rightarrow Z$, entonces $X \rightarrow YZ$
  - Descomposición: Si $X \rightarrow YZ$, entonces $X \rightarrow Y$ y $X \rightarrow Z$

Ejemplo: Clausura de un conjunto de DFs

- Consideremos el esquema de relación $R(A, B, C)$ con DFs: $A \rightarrow B$ y $B \rightarrow C$, aplicando los axiomas obtenemos D^+:
  - $A \rightarrow A$, $B \rightarrow B$, $C \rightarrow C$ por reflexividad
  - $A \rightarrow C$, por transitividad dado que $A \rightarrow B$ y $B \rightarrow C$
  - $AC \rightarrow BC$, por aumento de $A \rightarrow B$ con C
  - $AB \rightarrow AC$, por aumento de $B \rightarrow C$ con A
  - $AB \rightarrow CB$, por aumento de $A \rightarrow C$ con B

Formas Normales

- Las formas normales basadas en dependencias funcionales son:
  - Primera forma normal (1FN)
  - Segunda forma normal (2FN)
  - Tercera forma normal (3FN)
  - Forma normal Boyce-Codd (FNBC)
- Estas formas normales tienen requisitos cada vez más restrictivos
- Toda relación en FNBC está también 3FN, toda relación en 3FN está en 2FN y toda relación en 2FN está en 1FN

1FN y 2FN

- Una relación se encuentra en 1FN si todos los atributos contienen únicamente valores atómicos, i.e., no corresponden a listas ni conjuntos
- Una relación se encuentra en 2FN si está en 1FN y cada atributo no clave (o no primo) depende por completo de los atributos claves (o atributos primos)
- A los atributos que participan en claves se les denomina atributos primos y a los otros no primos

Ejemplo: Formas Normales

- Consideremos la relación $R(A, B, C, D)$ y las DFs:
  - $AB \rightarrow C$
  - $AB \rightarrow D$
  - $A \rightarrow C$
- ¿Está R en 2FN?
- Para responder esta pregunta primeros debemos saber cuales son las claves candidatas para R

Ejemplo: Formas Normales

- Consideremos la relación $R(A, B, C, D)$ y las DFs:
  - $AB \rightarrow C$
  - $AB \rightarrow D$
  - $A \rightarrow C$
- Existe una única clave candidata para R que es AB:
  - $AB \rightarrow C$ dada
  - $AB \rightarrow D$ dada
  - $AB \rightarrow A$ por reflexividad, dado que $A \subseteq AB$
  - $AB \rightarrow B$ por reflexividad, dado que $B \subseteq AB$
  - Atributos primos: $\{A, B\}$.
  - Atributos no primos: $\{C, D\}$.
- R no está en 2FN porque el atributo no primo C depende parcialmente de la clave primaria AB
- Debemos dividir R en dos relaciones $R_1(A, C)$ y $R_2(A, B, D)$

3FN

- Sea R el esquema de una relación, D el conjunto de DFs que se cumplen en R, X un subconjunto de los atributos de R y A un atributo de R
- R se encuentra en 3FN si, para cada DF en D de la forma $X \rightarrow A$, alguna de las afirmaciones siguientes es verdadera:
  1. $A \in X$, es decir, es una DF trivial o
  2. X es una super clave o
  3. A es parte de alguna clave de R (atributo primo)
- En otras palabras, una relación está en 3FN si está en 2FN y cada atributo no clave depende de manera no transitiva de atributos clave

Ejemplo: 3FN

- Consideremos el esquema

$$Horas\_Emp(id, nombre, depto, categoria, horas\_extras, horas\_trab)$$

y las DFs:

  - $id \rightarrow nombre, depto, categoria, horas\_extras, horas\_trab$
  - $categoria \rightarrow horas\_extras$

- ¿Está *Horas_Emp* en 3FN?
- Sabemos que la clave de esta relación es *id*

Ejemplo: 3FN

- Consideremos el esquema

  $$Horas\_Emp(id, nombre, depto, categoria, horas\_extras, horas\_trab)$$

  y las DFs:
  - $id \rightarrow nombre, depto, categoria, horas\_extras, horas\_trab$
  - $categoria \rightarrow horas\_extras$
- Para la primera DF
  $id \rightarrow nombre, depto, categoria, horas\_extras, horas\_trab$, id es una super clave
- Para la segunda DF $categoria \rightarrow horas\_extras$, $categoria$ no es super clave ni tampoco $horas\_extras$ es parte de alguna clave (no es atributo primo). Por lo tanto, $Horas\_Emp$ no está en 3FN
- Descomponemos en
  $Horas\_Emp_1(id, nombre, depto, categoria, horas\_trab)$ y
  $Horas\_Emp_2(categoria, horas\_extras)$

Ejemplo: 3FN

- Consideremos la relación $R(A, B, C, D, E)$ y las DFs:
  - $AB \rightarrow CDE$
  - $C \rightarrow E$
- ¿Está R en 3FN?
- Debemos determinar las claves candidatas para R

Ejemplo: 3FN

- Consideremos la relación $R(A, B, C, D, E)$ y las DFs:
  - $AB \rightarrow CDE$
  - $C \rightarrow E$
- La única clave para R es AB:
  - $AB \rightarrow CDE$ dada
  - $AB \rightarrow C$ por descomposición
  - $AB \rightarrow D$ por descomposición
  - $AB \rightarrow E$ por descomposición
  - $AB \rightarrow A$ por reflexividad, dado que $A \subseteq AB$
  - $AB \rightarrow B$ por reflexividad, dado que $B \subseteq AB$
  - Atributos primos: $\{A, B\}$
  - Atributos no primos: $\{C, D, E\}$
- R no está en 3FN porque C no es atributo super clave ni E es parte de una clave para R
- Debemos dividir R en dos relaciones $R_1(C, E)$ y $R_2(A, B, C, D)$

Ejemplo: 3FN

- Consideremos la relación $R(A, B, C, D, E)$ y las DFs:
  - $AB \rightarrow CDE$
  - $C \rightarrow A$
- ¿Está R en 3FN?

Ejemplo: 3FN

- Consideremos la relación $R(A, B, C, D, E)$ y las DFs:
  - $AB \rightarrow CDE$
  - $C \rightarrow A$
- Las claves para R son AB y BC.
- Demostramos AB:
  - $AB \rightarrow CDE$ dada
  - $AB \rightarrow C$ por descomposición
  - $AB \rightarrow D$ por descomposición
  - $AB \rightarrow E$ por descomposición
  - $AB \rightarrow A$ por reflexividad dado que $A \subseteq AB$
  - $AB \rightarrow B$ por reflexividad dado que $B \subseteq AB$

Ejemplo: 3FN

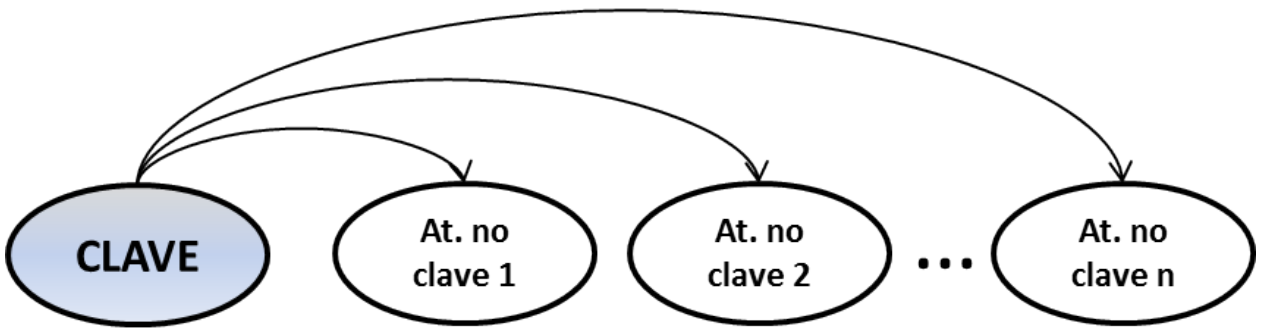
- Consideremos la relación $R(A, B, C, D, E)$ y las DFs:
  - $AB \rightarrow CDE$
  - $C \rightarrow A$
- Demostramos BC:
  - $C \rightarrow A$ dada
  - $BC \rightarrow AB$ aumentamos por B la DF: $C \rightarrow A$
  - $BC \rightarrow CDE$ por transitividad
  - $BC \rightarrow C$ por descomposición
  - $BC \rightarrow D$ por descomposición
  - $BC \rightarrow E$ por descomposición
- Atributos primos: $\{A, B, C\}$
- Atributos no primos: $\{D, E\}$
- R está en 3FN porque en la primera DF AB es super clave y en la segunda DF A es parte de una clave para R

Forma Normal Boyce-Codd

- Sea R el esquema de una relación, D el conjunto de DFs que se cumplen para R, X un subconjunto de los atributos de R y A un atributo de R
- R está en FNBC si, para cada DF en D de la forma $X \rightarrow A$ alguna de las siguientes afirmaciones se cumple:
  - $A \subseteq X$, i.e., $X \rightarrow A$ es una DF trivial o
  - X es una super clave
- De manera intuitiva, en las relaciones FNBC las únicas dependencias no triviales son aquellas en las que una clave determina algún atributo

Forma Normal Boyce-Codd

- Kent (1978) expresa *cada atributo debe describir la clave, toda la clave y nada más que la clave*
- Si se emplean óvalos para denotar atributos o conjuntos de atributos y se dibujan arcos para indicar las DFs, las relaciones en FNBC tienen la siguiente estructura:



Ejemplo: FNBC

- Consideremos la relación $R(A, B, C, D, E)$ y las DFs:
  - $AB \rightarrow CDE$
  - $C \rightarrow A$
- Las claves para R son AB y BC
- R está en 3FN porque en la primera DF AB es super clave, y en la segunda DF A es parte de una clave para R
- Sin embargo, R no está en FNBC por la segunda DF, ya que C no es superclave
- R no se puede descomponer

Ejemplo: FNBC

- Considere la relación $CodPostales(Ciudad, Calle, Cod)$ y las DFs:
  - $Ciudad, Calle \rightarrow Cod$
  - $Cod \rightarrow Ciudad$
- ¿Está *CodPostales* en FNBC?
- Necesitamos determinar las llaves candidatas para *CodPostales*

Ejemplo: FNBC

- Para *CodPostales*(*Ciudad*, *Calle*, *Cod*) con DFs:
  - *Ciudad*, *Calle* $\rightarrow$ *Cod*
  - *Cod* $\rightarrow$ *Ciudad*
- Las llaves candidatas son:
  - *Ciudad*, *Calle* (se obtiene directamente desde la primera DF)
  - *Cod*, *Calle* (se obtiene aumentando por *Calle* la segunda DF)
- Por lo tanto, *CodPostales* no está en FNBC, porque *Cod* no es una super clave (no contiene una llave candidata)
- *CodPostales* está en 3FN

Ejemplo: FNBC

- En general, una relación no está en FNBC cuando existen dos o más claves candidatas para la relación, éstas son claves compuestas (con más de un atributo) y su intersección no es vacía
- Para $CodPostales(Ciudad, Calle, Cod)$ con DFs:
  - $Ciudad, Calle \rightarrow Cod$
  - $Cod \rightarrow Ciudad$
- Existen dos claves candidatas que son compuestas:
  - *Ciudad, Calle*
  - *Cod, Calle*
- La intersección de ellas no es vacía (*Calle*)

Algoritmo Descomposición en FNBC

- Supóngase una relación R con un conjunto D de DFs que no está en FNBC:
  1. Sea $X \subset R$, A un atributo de R y $X \rightarrow A$ una DF que provoca una violación de la FNBC
  2. Descomponer R en $R - A$ y XA
  3. Si $R - A$ o XA no están en FNBC, descomponer de nuevo mediante la aplicación recursiva de este algoritmo

Ejemplo: Algoritmo Descomposición en FNBC

- Consideremos el esquema $Contratos(C, P, Y, D, R, N, V)$ y las DFs:
  - $C \rightarrow CPYDRNV$
  - $YR \rightarrow C$
  - $PD \rightarrow R$
  - $Y \rightarrow P$
- Las claves candidatas son: C e YR
- ¿Está la relación en FNBC?

Ejemplo: Algoritmo Descomposición en FNBC

- Consideremos el esquema $Contratos(C, P, Y, D, R, N, V)$ y las DFs:
  - $C \rightarrow CPYDRNV$
  - $YR \rightarrow C$
  - $PD \rightarrow R$
  - $Y \rightarrow P$
- La relación no está en FNBC porque en las dependencias funcionales $PD \rightarrow R$ e $Y \rightarrow P$, ni PD ni tampoco Y son **claves**
- *Contratos* se descompone en $C_1(P, D, R)$ y $C_2(C, P, Y, D, N, V)$
- $C_1(P, D, R)$ está en FNBC, pero $C_2(C, P, Y, D, N, V)$ no, porque en la DF $Y \rightarrow P$, Y no es **clave**
- Por lo tanto, $C_2(C, P, Y, D, N, V)$ se divide en $C_3(Y, P)$ y $C_4(C, Y, D, N, V)$
- Finalmente, $C_1(P, D, R)$, $C_3(Y, P)$ y $C_4(C, Y, D, N, V)$ están todas en FNBC

Propiedades de las Descomposiciones

- La descomposición es una herramienta que permite eliminar la redundancia
- Sin embargo, es importante comprobar que la descomposición no introduzca nuevos problemas
- En especial conviene comprobar que:
  1. La descomposición permite recuperar la relación original
  2. La descomposición permite comprobar de modo eficiente las restricciones de integridad

Descomposición por Join sin Pérdida

- Sea R el esquema de una relación y sea F un conjunto de DFs en R
- Una descomposición de R en dos esquemas con los conjuntos de atributos X e Y es una descomposición por join sin pérdida con respecto a F si, para todos los ejemplares r de R que satisfacen las dependencias funcionales de F:

$$\Pi_X(r) \bowtie \Pi_Y(r) = r$$

- En otras palabras, se puede recuperar la relación original a partir de las relaciones descompuestas
- Todas las descomposiciones empleadas para eliminar la redundancia deben ser sin pérdida

Ejemplo: Descomposición sin Pérdida

- Consideremos la relación R y las DFs: $S \rightarrow P$ y $P \rightarrow D$
- La siguiente descomposición de R en R_1 y R_2 es por join sin pérdida porque $R = R_1 \bowtie R_2$:

R		
S	P	D
s_1	100	d_1
s_2	200	d_2
s_3	300	d_3

R_1	
S	P
s_1	100
s_2	200
s_3	300

R_2	
P	D
100	d_1
200	d_2
300	d_3

$R_1 \bowtie R_2$		
S	P	D
s_1	100	d_1
s_2	200	d_2
s_3	300	d_3

Ejemplo: Descomposición con Pérdida

- La instancia original viola la DF: $P \rightarrow D$
- Por lo tanto, la siguiente descomposición es incorrecta dado que $R \neq R_1 \bowtie R_2$:

R		
S	P	D
s_1	100	d_1
s_2	200	d_2
s_3	300	d_3
s_4	200	d_4

R_1	
S	P
s_1	100
s_2	200
s_3	300
s_4	200

R_2	
P	D
100	d_1
200	d_2
300	d_3
200	d_4

$R_1 \bowtie R_2$		
S	P	D
s_1	100	d_1
s_2	200	d_2
s_2	200	d_4
s_3	300	d_3
s_4	200	d_2
s_4	200	d_4

Descomposiciones que Conservan las Dependencias

- Al descomponer una relación podríamos perder algunas DFs y más aún a veces es imposible obtener un conjunto de relaciones que conserve todas las dependencias originales
- Consideremos el esquema $Contratos(C, P, Y, D, R, N, V)$ y las DFs: $C \rightarrow CPYDRNV$, $YR \rightarrow C$, $PD \rightarrow R$
  - Las claves candidatas son: C e YR
  - La relación no está en FNBC porque PD no es clave
  - $Contratos$ se descompone en $C_1(C, P, Y, D, N, V)$ y $C_2(P, D, R)$
  - Ya no podemos mantener la dependencia $YR \rightarrow C$ sin tener que hacer un join entre C_1 y C_2
- Esta descomposición no conserva todas las dependencias funcionales