

Bases de Datos

Clase 7: Dependencias y Formas Normales

Buen diseño de relaciones

- Tenemos nuestro E/R
- Lo sabemos transformar a un esquema relacional
- ¿Siempre estará todo perfecto?

Redundancia en los datos

Quizás queremos guardar información de:

- Guías de una agencia de turismo
- Necesitamos su id y nombre
- Número de horas que trabajaron
- Los turistas le ponen un score/evaluación
- Pago por hora depende del score

Redundancia en los datos

Quizás queremos guardar información de:

- Guías de una agencia de turismo
- Necesitamos su id y nombre
- Número de horas que trabajaron
- Los turistas le ponen un score/evaluación
- Pago por hora depende del score

Guías(gid, nombre, score, horas, valorHora)

Redundancia en los datos

Guías

gid	nombre	score	horas	valorHora
1	Juan	8	40	18000
2	Johanna	8	30	18000
3	Cristian	5	30	15000
4	Pedro	8	32	18000

Problemas con la redundancia

Gasto de espacio

Guías

gid	nombre	score	horas	valorHora
1	Juan	8	40	18000
2	Johanna	8	30	18000
3	Cristian	5	30	15000
4	Pedro	8	32	18000

Guardamos el valor 18000 muchas veces

Problemas con la redundancia

Anomalías de inserción

Guías

gid	nombre	score	horas	valorHora
1	Juan	8	40	18000
2	Johanna	8	30	18000
3	Cristian	5	30	15000
4	Pedro	8	32	18000
5	Pablo	9	34	???

Si conocemos el score, pero no el valorHora,
no podemos insertar este guía a la tabla

Problemas con la redundancia

Anomalías de actualización

Guías

gid	nombre	score	horas	valorHora
1	Juan	8	40	17000
2	Johanna	8	30	18000
3	Cristian	5	30	15000
4	Pedro	8	32	18000

Si cambia el valor hora para el score 8

Problemas con la redundancia

Anomalías de actualización

Guías

gid	nombre	score	horas	valorHora
1	Juan	8	40	17000
2	Johanna	8	30	18000
3	Cristian	5	30	15000
4	Pedro	8	32	18000

Si cambia el valor hora para el score 8

Datos inconsistentes!!

Problemas con la redundancia

Anomalías de actualización

Guías

gid	nombre	score	horas	valorHora
1	Juan	8	40	17000
2	Johanna	8	30	17000
3	Cristian	5	30	15000
4	Pedro	8	32	17000

Si cambia el valor hora para el score 8

Cambiar en todas las tuplas

Problemas con la redundancia

Anomalías de eliminación

Guías

gid	nombre	score	horas	valorHora
1	Juan	8	40	17000
2	Johanna	8	30	17000
3	Cristian	5	30	15000
4	Pedro	8	32	17000

Si elimino a Cristian pierdo información

Solución

Descomponer la tabla

Guías

gid	nombre	score	horas	valorHora
1	Juan	8	40	17000
2	Johanna	8	30	17000
3	Cristian	5	30	15000
4	Pedro	8	32	17000

Solución

Descomponer la tabla

Guías

gid	nombre	score	horas
1	Juan	8	40
2	Johanna	8	30
3	Cristian	5	30
4	Pedro	8	32

Valor

score	valorHora
8	18000
5	15000

Solución

¿Qué pasó aquí?

- **Score** determina **valorHora**

Esto se llama una **dependencia funcional**

Restricciones de Integridad

- Los datos deben satisfacer restricciones de integridad
- Estas restricciones son importantes en la modelación
- ¿Cómo nos pueden ayudar a especificar lo que queremos en cada relación?

Dependencia Funcional

Una dependencia funcional en la relación R es:

$$X \rightarrow Y$$

Dónde X e Y son conjuntos de atributos de la relación R

Dependencia Funcional

Definición

$X \rightarrow Y$ es válida en una relación R ssi para toda tupla $t_1, t_2 \in R$ se tiene:

$$\pi_X(t_1) = \pi_X(t_2) \quad \text{implica} \quad \pi_Y(t_1) = \pi_Y(t_2)$$

Dependencia Funcional

Ejemplo

Guías(gid, nombre, score, horas, valorHora)

- $\text{score} \rightarrow \text{valorHora}$

Personas(rut, nombre)

- $\text{rut} \rightarrow \text{nombre}$

Películas(pid, título, año, director)

- $\text{título, año} \rightarrow \text{director}$

Dependencia Funcional

Ejemplo

Guías(gid, nombre, score, horas, valorHora)

- $\text{score} \rightarrow \text{valorHora}$

Personas(rut, nombre)

- $\text{rut} \rightarrow \text{nombre}$

Películas(pid, título, año, director)

- $\text{título, año} \rightarrow \text{director}$



Lado izquierdo no es necesariamente una llave

Dependencia Funcional

Ejemplo

Programación(cine, película, director, dirección, teléfono,
horario, precio)

- cine → teléfono, dirección
- película → director
- cine, película, horario → precio

¿Cuál va a ser la llave?

Buenas y malas dependencias

DJE

Depto	Jefe	Empleado
D1	Pérez	Ureta
D1	Pérez	Assad
D2	Correa	Vargas
D1	Pérez	Gómez
D1	Pérez	Camus
...

ES

Empleado	Salario
Ureta	600
Assad	800
Vargas	800
...	...

Una dependencia buena

- **DJE:** Depto \rightarrow Jefe
- **ES:** Empleado \rightarrow Salario (Empleado es llave)

Buenas y malas dependencias

Anomalía de inserción

DJE

Depto	Jefe	Empleado
D1	Pérez	Ureta
D1	Pérez	Assad
D2	Correa	Vargas
D1	Pérez	Gómez
D1	Pérez	Camus
...

Compañía contrata a un empleado, pero no lo asigna a un departamento

No podemos almacenarlo en **DJE**

Buenas y malas dependencias

Anomalía de eliminación

DJE

Depto	Jefe	Empleado
D1	Pérez	Ureta
D1	Pérez	Assad
D2	Correa	Vargas
D1	Pérez	Gómez
D1	Pérez	Camus
...

El empleado Vargas abandona la empresa, por lo que hay que eliminarlo de **DJE**

¡Al hacer eso eliminamos también al jefe Correa!

Buenas y malas dependencias

Redundancia

DJE

Depto	Jefe	Empleado
D1	Pérez	Ureta
D1	Pérez	Assad
D2	Correa	Vargas
D1	Pérez	Gómez
D1	Pérez	Camus
...

Tenemos dos tuplas indicando que Pérez es jefe de D1

Buenas y malas dependencias

El problema es que la asociación entre jefes y empleados se almacena en la misma tabla que la asociación entre jefes y departamentos

También el mismo hecho puede ser almacenado muchas veces, como que jefe está a cargo de que departamento (ej. Pérez con D1)

Buenas y malas dependencias

Existe dependencia Depto → Jefe
pero Depto **no es llave**

¡Este tipo de situaciones queremos evitar!

¡Porque pueden causar **redundancia**!

Anomalías

Objetivo: Eliminar anomalías tratando de minimizar la redundancia, para esto:

- Debemos averiguar las dependencias que aplican
- Descomponer las tablas en tablas más pequeñas

Anomalías

Objetivo: Eliminar anomalías tratando de minimizar la redundancia, para esto:

- **Debemos averiguar las dependencias que aplican**
- Descomponer las tablas en tablas más pequeñas

Dependencias

Observación 1

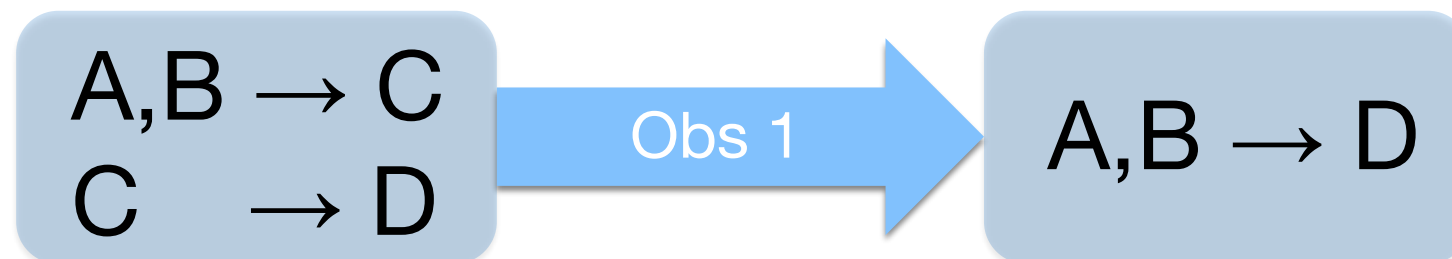
Si tenemos las dependencias:

- $X \rightarrow Y$
- $Y \rightarrow Z$

Podemos deducir que:

- $X \rightarrow Z$

Con X, Y, Z conjuntos de atributos



Dependencias

Observación 2

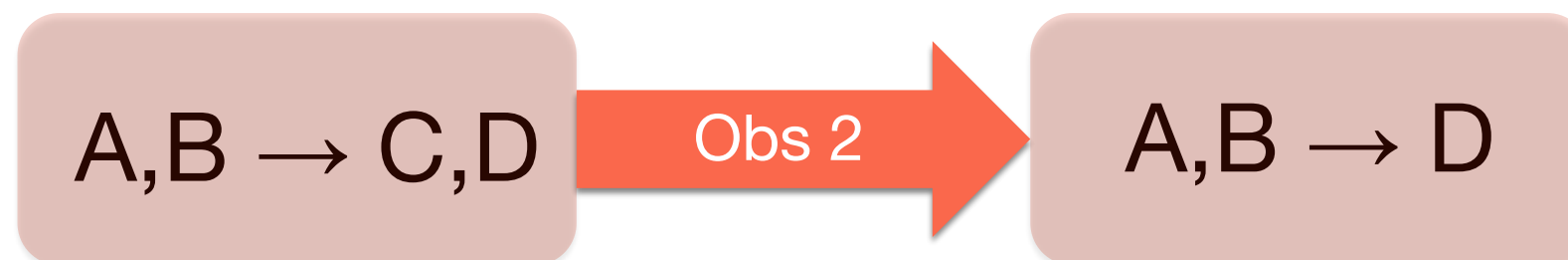
Si $\mathbf{Z} \subseteq \mathbf{Y}$, y tenemos la dependencia:

- $X \rightarrow Y$

Podemos deducir que:

- $X \rightarrow Z$

Con X, Y, Z conjuntos de atributos



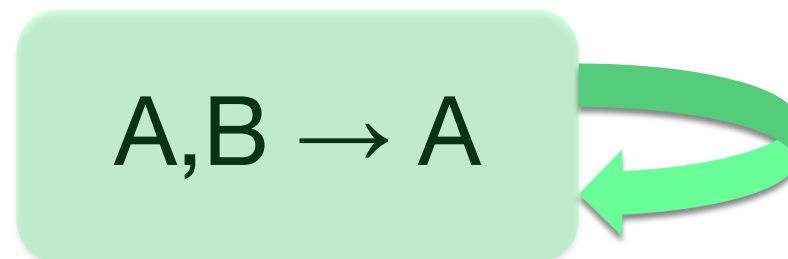
Dependencias

Observación 3

Llamamos dependencia trivial a la dependencia:

$$X \rightarrow Y$$

Si se tiene que $Y \subseteq X$



Dependencias

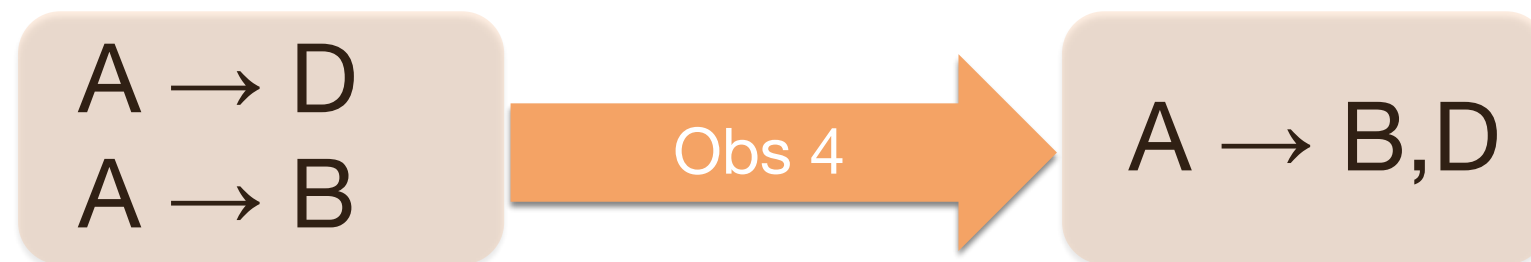
Observación 4

Si tenemos que:

$$X \rightarrow Y, X \rightarrow Z$$

Podemos decir que:

$$X \rightarrow Y, Z$$



Dependencias

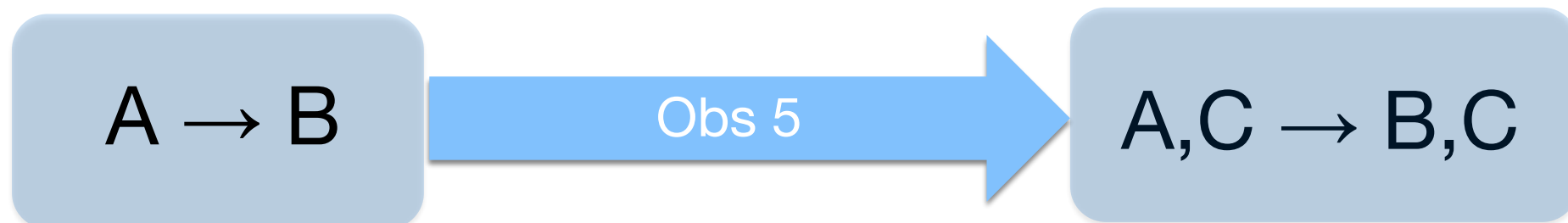
Observación 5

Si tenemos que:

$$X \rightarrow Y$$

Podemos decir que para cada Z:

$$X, Z \rightarrow Y, Z$$



Dependencias

Observación 6

Si tenemos $X \rightarrow Y$, los atributos **X** son (candidatos a) llave si **Y** contiene a todos los atributos que son parte de la relación y no están en **X**

$R(A,B,C,D,E)$



Observación 6: A,B es llave candidata

Dependencias

Ejemplo

Averiguar todas las dependencias de $R(a, b, c)$ si:

- $a \rightarrow b$
- $b \rightarrow a, c$

Podemos inferir además $a \rightarrow b, c$
(por lo tanto, **a** es llave)

Dependencias

Ejemplo

Podemos inferir además $a \rightarrow b, c$
(por lo tanto, **a** es llave)

Demostración: supongo que para tuplas t_1, t_2 tengo

$$\pi_a(t_1) = \pi_a(t_2)$$

Como tengo $a \rightarrow b$, se cumple que $\pi_b(t_1) = \pi_b(t_2)$

Dependencias

Ejemplo

Pero $b \rightarrow a, c$, luego $\pi_{a,c}(t_1) = \pi_{a,c}(t_2)$

Finalmente $\pi_c(t_1) = \pi_c(t_2)$

Importante: con esta idea podemos demostrar que nuestras observaciones 1 — 6 son correctas

Dependencias

Ejercicio

Averiguar todas las dependencias:

Toma(alumno, carrera, ramo, sala, hora)

- alumno \rightarrow carrera
- carrera, ramo \rightarrow sala
- ramo \rightarrow hora

Dependencias

Ejercicio

Averiguar todas las dependencias:

Toma(alumno, carrera, ramo, sala, hora)

- alumno \rightarrow carrera
- carrera, ramo \rightarrow sala
- ramo \rightarrow hora

alumno, ramo \rightarrow carrera, sala, hora

Anomalías

Objetivo: Eliminar anomalías tratando de minimizar la redundancia, para esto:

- Debemos averiguar las dependencias que aplican
- **Descomponer las tablas en tablas más pequeñas**

Descomposición

Ejemplo: mal diseño

Información: cine, película, director, dirección, teléfono, horario, precio

- cine → dirección, teléfono
- película → director
- cine, película, horario → precio

El peor diseño:

MAL(cine, película, director, dirección, teléfono, horario, precio)

Descomposición

Ejemplo: mal diseño

MAL(cine, película, director,dirección, teléfono, horario, precio)

Redundancia:

- La película determina al director, pero cada vez que dan la película los listamos a ambos
- Listamos la dirección y el teléfono del cine una y otra vez

Descomposición

Ejemplo: mal diseño

MAL(cine, película, director, dirección, teléfono, horario, precio)

Anomalías:

- Si cambiamos una dirección nos volvemos inconsistentes, hay que cambiarla en todas las tuplas
- Si dejamos de mostrar una película perdemos la asociación director - película
- No podemos agregar películas que no se muestran

Descomposición

Ejemplo: buen diseño

Dividimos MAL en 3 tablas

Rels:	Atributos	Dependencias
R_1	cine, dirección, teléfono	cine \rightarrow dirección, teléfono
R_2	cine, película, horario, precio	cine, película, horario \rightarrow precio
R_3	película, director	película \rightarrow director

Descomposición

Ejemplo: buen diseño

Es un buen diseño porque:

- No hay anomalías
- No perdemos dependencias funcionales, pues todas están restringidas a sus respectivas tablas
- No perdemos información:

$$R_1 = \pi_{cine,direccion,telefono}(MAL)$$

$$R_2 = \pi_{cine,pelicula,horario,precio}(MAL)$$

$$R_3 = \pi_{pelicula,director}(MAL)$$

$$MAL = R_1 \bowtie R_2 \bowtie R_3$$

¿Cómo lograr esto para un esquema complejo?

Llaves (repaso)

Super llave (superkey): cualquier conjunto de atributos que determina a todo el resto

Llave (candidata/minimal): cualquier conjunto de atributos que determina a todo el resto, y ninguno de sus subconjuntos es una super llave

Llave primaria: una llave candidata que queremos destacar

Surrogate Keys

Alumnos(aid, rut, nombre, apellido, carrera, correo)

Llaves:

- **aid**
- **rut**
- **correo**

Super llaves:

- **aid**
- **rut**
- **correo**
- **aid, nombre**
- **rut, apellido, carrera**
- **...**

Boyce-Codd Normal Form (BCNF)

Causa de anomalías: $X \rightarrow Y$ cuando X no es una super llave

Una relación **R** está en **BCNF** si para toda dependencia funcional no trivial $X \rightarrow Y$, **X** es una super llave

Un esquema está en BCNF si todas sus relaciones están en BCNF

¿Cuándo estoy en BCNF?

Alumnos(aid, rut, nombre, apellido, carrera, correo)

Llaves:

- **aid**
- **rut**
- **correo**

Dependencias:

- **aid -> todo**
- **rut -> todo**
- **correo -> todo**
- **aid,rut -> todo**
- **aid,nombre -> todo**

...



superllaves

Basta concentrarnos en las llaves! -- ¿Por qué?

¿Cuándo estoy en BCNF?

Para BCNF hay que pescar las dependencias:

- llave → resto (ojo: llave candidata, posiblemente más de una)
- notSuperLlave → algo (ya no estoy en BCNF)

AlumnoTelefono(rut, teléfono, nombre, apellido):

- rut, telefono -> todo (esto es la única llave, todo OK)
- rut -> nombre, apellido (como rut no es superLlave, entonces el esquema no está en BCNF)

BCNF

¿Cómo lograr BCNF?

BCNF se logra mediante descomposiciones

Ya vimos una: de MAL a tres tablas

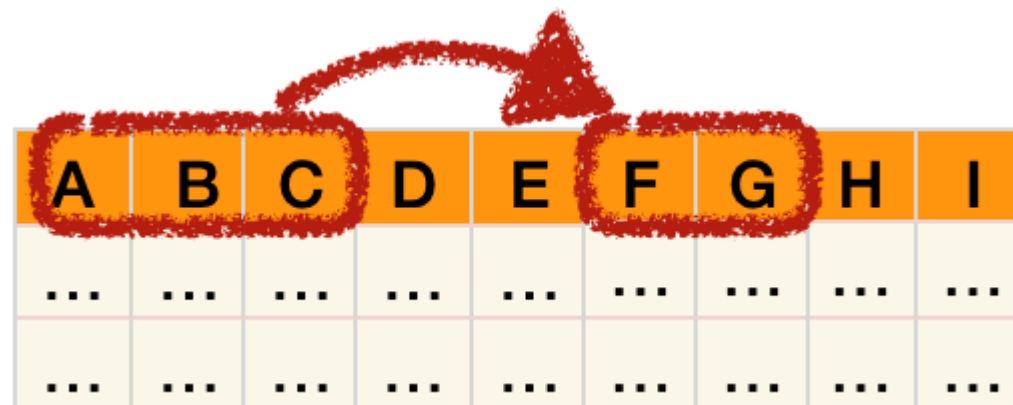
BCNF

Algoritmo

INPUT: - el esquema de la tabla $R(A_1, \dots, A_n)$ con atributos A_1, \dots, A_n
- dependencias funcionales estandarizadas

1. Tomar una dependencia funcional $X \rightarrow Y$, dónde X no es una super llave y descomponer en 2 tablas con esquemas:

$R1(X, Y)$ y $R2(\{A_1, \dots, A_n\} - Y)$



A	B	C	D	E	F	G	H	I
...
...

$A, B, C \rightarrow F, G$

BCNF

Algoritmo

INPUT: - el esquema de la tabla $R(A_1, \dots, A_n)$ con atributos A_1, \dots, A_n
- dependencias funcionales estandarizadas

1. Tomar una dependencia funcional $X \rightarrow Y$, dónde X no es una super llave y descomponer en 2 tablas con esquemas:

$R1(X, Y)$ y $R2(\{A_1, \dots, A_n\} - Y)$

A	B	C	F	G
...
...

A	B	C	D	E	H	I
...
...

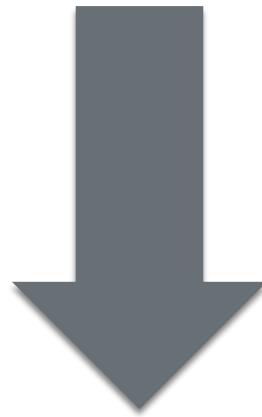
2. Repetir con las tablas que aun no están en BCNF

BCNF

Algoritmo

S(a,b,c,d,e,f)

a,b \rightarrow c,e rompe BCNF



S1(a,b,c,e)

S2(a,b,d,f)

BCNF

Algoritmo

$$x_1, \dots, x_n \rightarrow y_1, \dots, y_k$$

Se repite el proceso anterior hasta que no haya más violaciones de BCNF

Optimización: elegir la mayor cantidad de B's posibles

BCNF

Ejemplo

¿Cómo descomponemos **R** para lograr BCNF?

$R(a, b, c, d)$

$a \rightarrow b, b \rightarrow c$

Podemos deducir $a \rightarrow c, a \rightarrow b, c$

BCNF

Ejemplo

Descomposición 1:

$R(a, b, c, d)$

- $a \rightarrow b$
- $b \rightarrow c$
- $a \rightarrow c$
- $a \rightarrow b, c$

$R(a, b, c, d)$

BCNF

Ejemplo

Descomposición 1:

$R(a, b, c, d)$

- $a \rightarrow b$
- $b \rightarrow c$
- $a \rightarrow c$
- $a \rightarrow b, c$



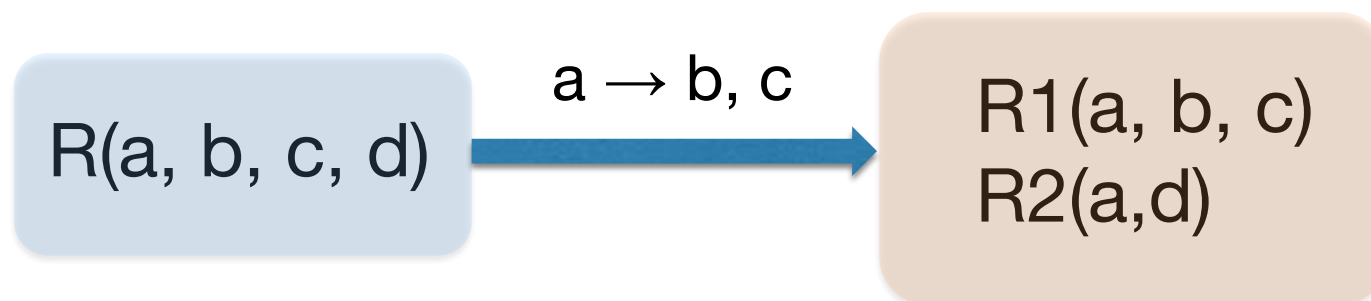
BCNF

Ejemplo

Descomposición 1:

$R(a, b, c, d)$

- $a \rightarrow b$
- $b \rightarrow c$
- $a \rightarrow c$
- $a \rightarrow b, c$



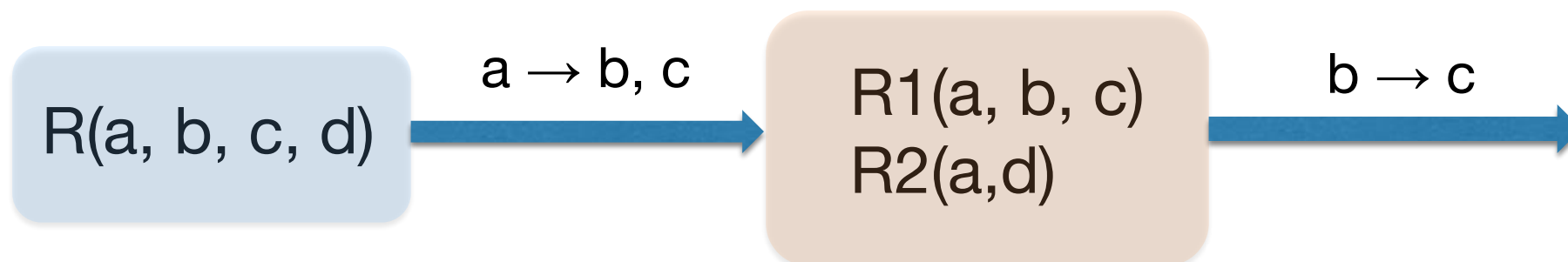
BCNF

Ejemplo

Descomposición 1:

$R(a, b, c, d)$

- $a \rightarrow b$
- $b \rightarrow c$
- $a \rightarrow c$
- $a \rightarrow b, c$



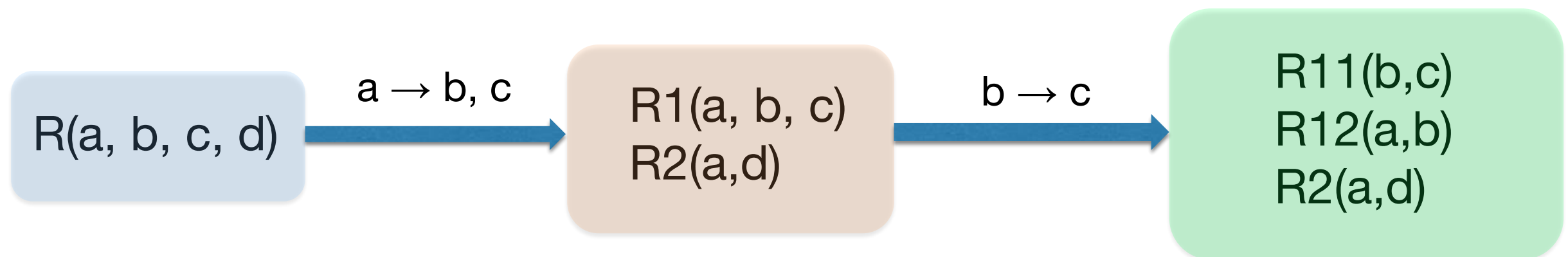
BCNF

Ejemplo

Descomposición 1:

$R(a, b, c, d)$

- $a \rightarrow b$
- $b \rightarrow c$
- $a \rightarrow c$
- $a \rightarrow b, c$



BCNF

Ejemplo

Descomposición 2:

$R(a, b, c, d)$

- $a \rightarrow b$
- $b \rightarrow c$
- $a \rightarrow c$
- $a \rightarrow b, c$



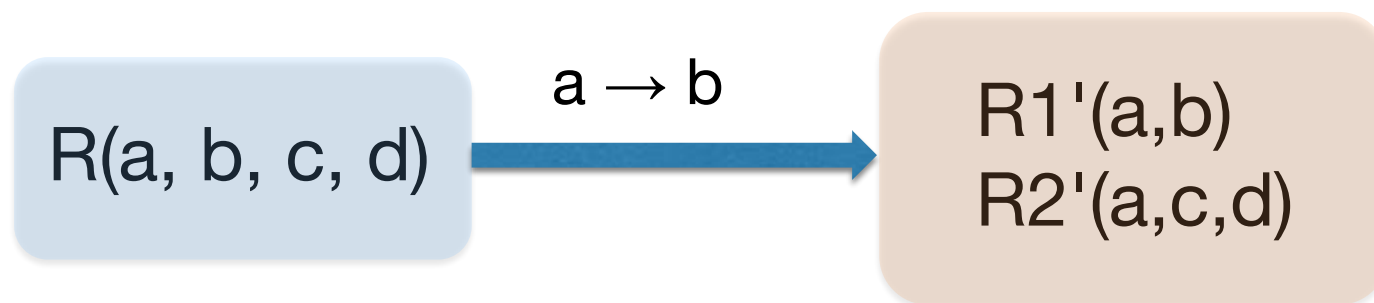
BCNF

Ejemplo

Descomposición 2:

$R(a, b, c, d)$

- $a \rightarrow b$
- $b \rightarrow c$
- $a \rightarrow c$
- $a \rightarrow b, c$



BCNF

Ejemplo

Descomposición 2:

$R(a, b, c, d)$

- $a \rightarrow b$
- $b \rightarrow c$
- $a \rightarrow c$
- $a \rightarrow b, c$



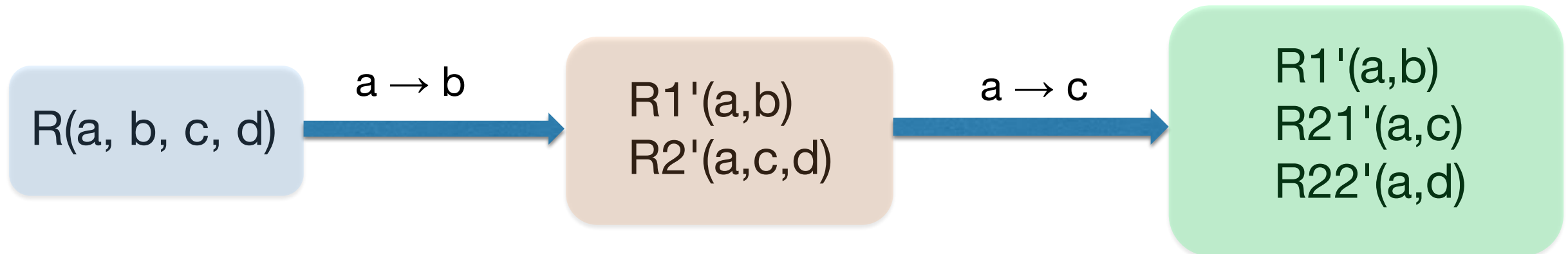
BCNF

Ejemplo

Descomposición 2:

$R(a, b, c, d)$

- $a \rightarrow b$
- $b \rightarrow c$
- $a \rightarrow c$
- $a \rightarrow b, c$



BCNF

Ejemplo

Descomposición 1:

R11(b,c)
R12(a,b)
R2(a,d)

Descomposición 2:

R1'(a,b)
R21'(a,c)
R22'(a,d)

BCNF

Ejemplo

Descomposición 1:

R11(b,c)
R12(a,b)
R2(a,d)

Descomposición 2:

R1'(a,b)
R21'(a,c)
R22'(a,d)

Pero: el join natural es equivalente

Pérdida de Información

La descomposición no puede perder información!

Producto	nombre	precio	categoría
	Canon T3	300	fotografía
	Nokia 5000	400	fotografía
	Galaxy IV	400	celular

nombre	categoría	precio	categoría
Canon T3	fotografía	300	fotografía
Nokia 5000	fotografía	400	fotografía
Galaxy IV	celular	400	celular

Pérdida de Información

La descomposición no puede perder información!

Al hacer el join:

Producto	nombre	precio	categoría
	Canon T3	300	fotografía
	Canon T3	400	fotografía
	Nokia 5000	300	fotografía
	Nokia 5000	400	fotografía
	Galaxy IV	400	celular

Descomposición sin pérdida

$R(A, B, C)$ descompuesta en $R_1(A, B)$ y $R_2(A, C)$ es sin pérdida de información si para toda instancia de R :

$$R_1 \bowtie R_2 = R$$

Descomposición sin pérdida

Teorema

Para todo esquema con relación $R(X, Y, Z)$ y dependencia funcional $X \rightarrow Y$, para X, Y, Z conjuntos de atributos disjuntos, se tiene que la descomposición en $R1(X, Y)$ y $R2(X, Z)$ con $X \rightarrow Y$ es **sin pérdida de información**

Problemas con BCNF

Nuestra descomposición siempre va a ser sin pérdida de información, sin embargo puede ocurrir lo siguiente:

$R(a,b,c,d,e)$

- DF1: $a,b,c \rightarrow d,e$
- DF2: $d,e \rightarrow b$

llaves:

- a,b,c
- d,e,a,c

¡Hay una violación de BCNF: $d,e \rightarrow b$!

Problemas con BCNF

¡Hay que descomponer usando DF2!

$R(a,b,c,d,e)$

- DF1: $a,b,c \rightarrow d,e$
- DF2: $d,e \rightarrow b$

Descomposición



$R1(d,e,b)$

$R2(a,c,d,e)$

Observaciones:

- Tabla R1 permite validar que se cumple DF2
- No hay una tabla que permite validar DF1!

Problemas con BCNF

¡Hay que descomponer usando DF2!

$R(a,b,c,d,e)$

- DF1: $a,b,c \rightarrow d,e$
- DF2: $d,e \rightarrow b$

Descomposición



$R1(d,e,b)$

$R2(a,c,d,e)$

Observaciones:

- Tabla R1 permite validar que se cumple DF2
- No hay una tabla que permite validar DF1!
- Hay que hacer el join de R1 y R2 para validar DF1!

3NF

Una relación **R** está en **3NF** si para toda dependencia funcional no trivial $X \rightarrow Y$, **X** es una superllave o **Y** es parte de una llave minimal

Z es llave minimal si no existe llave **Z'** tal que $Z' \subseteq Z$

3NF es menos restrictivo que BCNF ya que permite un poco más de redundancia

BCNF vs 3NF

Nuestra descomposición siempre va a ser sin pérdida de información, sin embargo puede ocurrir lo siguiente:

$R(a,b,c,d,e)$

- DF1: $a,b,c \rightarrow d,e$
- DF2: $d,e \rightarrow b$

llaves:

- a,b,c
- d,e,a,c

¡Hay una violación de BCNF: $d,e \rightarrow b$!

¡Pero el esquema ya está en 3NF!

3NF

Algoritmo (idea)

INPUT:

- el esquema de la tabla **R**
- dependencias funcionales en un formato estandarizado
- sin dependencias inútiles (no lo especificamos bien)

1. Para **cada** dependencia funcional $X \rightarrow Y$ crear una tabla con esquema $X \cup Y$
2. Si al final, los esquemas resultantes R_1, \dots, R_n no contienen una llave de la tabla **R**, agregar una.

3NF

Algoritmo (idea)

IDEA: lado izquierdo mínimo;
remover dependencias que
se pueden deducir!

INPUT:

- el esquema de la tabla **R**
- dependencias funcionales en un formato estandarizado
- sin dependencias inútiles (**no lo especificamos bien**)

1. Para **cada** dependencia funcional $X \rightarrow Y$ crear una tabla con esquema $X \cup Y$
2. Si al final, los esquemas resultantes R_1, \dots, R_n no contienen una llave de la tabla **R**, agregar una.

3NF

Algoritmo (idea)

¡Solo para una tabla!

INPUT:

- el esquema de la tabla **R**
- dependencias funcionales en un formato estandarizado
- sin dependencias inútiles (no lo especificamos bien)

1. Para **cada** dependencia funcional $X \rightarrow Y$ crear una tabla con esquema $X \cup Y$
2. Si al final, los esquemas resultantes R_1, \dots, R_n no contienen una llave de la tabla **R**, agregar una.

3NF

Algoritmo (idea)

¡Solo para una tabla!

INPUT:

- el esquema de la tabla **R**
- dependencias funcionales en un formato estandarizado
- sin dependencias inútiles (no lo especificamos bien)

La mitad de lo que hace un paso de BCNF

1. Para **cada** dependencia funcional $X \rightarrow Y$ crear una tabla con esquema $X \cup Y$
2. Si al final, los esquemas resultantes R_1, \dots, R_n no contienen una llave de la tabla **R**, agregar una.

3NF

Algoritmo (idea)

INPUT:

- el esquema de la tabla **R**
- dependencias funcionales en un formato estandarizado
- sin dependencias inútiles (no lo especificamos bien)

IDEA: esto me asegura las dependencias!
(Cada una en su tabla)

1. Para **cada** dependencia funcional $X \rightarrow Y$ crear una tabla con esquema $X \cup Y$
2. Si al final, los esquemas resultantes R_1, \dots, R_n no contienen una llave de la tabla **R**, agregar una.

IDEA: esto me asegura que no hay pérdida de información!

3NF

Ejemplo

1. Para cada df $X \rightarrow Y$ crear una tabla con esquema $X \cup Y$

<u>Nombre Comuna</u>	<u>Región</u>	Tasa x m2	RUT	Nombre	Apellido	<u>Rol Lote</u>	M2	Avaluo
A	I	2	111111	Claudio	Gonzalez	34	455	960
A	I	2	111111	Claudio	Gonzalez	35	570	1040
A	I	2	222222	Maria	Zapata	27	895	1790
B	III	1,1	111111	Claudio	Gonzalez	10	150	165
B	III	1,1	333333	Carlos	Fernandez	11	200	220
B	X	1,1	444444	Elena	Abarca	13	150	165
C	V	0,5	555555	Luisa	Muñoz	2	500	250
D	V	3,5	111111	Claudio	Gonzalez	11	100	350

NombreComuna, Region \rightarrow Tasa

RUT \rightarrow Nombre, Apellido

NombreComuna, Region, Rol \rightarrow RUT, M2

Tasa, m2 \rightarrow Avalúo



{NombreComuna, Region, Rol }

3NF

Ejemplo

1. Para cada df $X \rightarrow Y$ crear una tabla con esquema $X \cup Y$

<u>Nombre Comuna</u>	<u>Región</u>	Tasa x m2
A	I	2
A	I	2
B	III	1,1
B	X	1,1
C	V	0,5

NombreComuna, Region \rightarrow Tasa

RUT \rightarrow Nombre, Apellido

NombreComuna, Region, Rol \rightarrow RUT, M2

Tasa, m2 \rightarrow Avalúo



{NombreComuna, Region, Rol }

3NF

Ejemplo

1. Para cada df $X \rightarrow Y$ crear una tabla con esquema $X \cup Y$

<u>Nombre Comuna</u>	<u>Región</u>	Tasa x m2
A	I	2
A	I	2
B	III	1,1
B	X	1,1
C	V	0,5

<u>RUT</u>	Nombre	Apellido
111111	Claudio	Gonzalez
222222	Maria	Zapata
333333	Carlos	Fernandez
...

$RUT \rightarrow \text{Nombre, Apellido}$

$\text{NombreComuna, Region} \rightarrow \text{Tasa}$

$\text{NombreComuna, Region, Rol} \rightarrow \text{RUT, M2}$
 $\text{Tasa, m2} \rightarrow \text{Avalúo}$



{NombreComuna, Region, Rol }

3NF

Ejemplo

1. Para cada df $X \rightarrow Y$ crear una tabla con esquema $X \cup Y$

<u>Nombre Comuna</u>	<u>Región</u>	Tasa x m2
A	I	2
A	I	2
B	III	1,1
B	X	1,1
C	V	0,5

NombreComuna, Region \rightarrow Tasa

<u>RUT</u>	Nombre	Apellido
111111	Claudio	Gonzalez
222222	Maria	Zapata
333333	Carlos	Fernandez
...

RUT \rightarrow Nombre, Apellido

<u>Tasa x m2</u>	<u>M2</u>	Avaluo
34	455	960
35	570	1040
27	895	1790
...

Tasa, m2 \rightarrow Avalúo

NombreComuna, Region, Rol \rightarrow RUT, M2



{NombreComuna, Region, Rol }

3NF

Ejemplo

1. Para cada df $X \rightarrow Y$ crear una tabla con esquema $X \cup Y$

<u>Nombre Comuna</u>	<u>Región</u>	Tasa x m2
A	I	2
A	I	2
B	III	1,1
B	X	1,1
C	V	0,5

NombreComuna, Region \rightarrow Tasa

<u>RUT</u>	Nombre	Apellido
111111	Claudio	Gonzalez
222222	Maria	Zapata
333333	Carlos	Fernandez
...

RUT \rightarrow Nombre, Apellido

<u>Tasa x m2</u>	<u>M2</u>	Avaluo
34	455	960
35	570	1040
27	895	1790
...

Tasa, m2 \rightarrow Avalúo

<u>Nombre Comuna</u>	<u>Región</u>	RUT	<u>Rol Lote</u>	M2
A	I	111111	34	455

NombreComuna, Region, Rol \rightarrow RUT, M2



{NombreComuna, Region, Rol }

3NF

Ejemplo

2. Si al final, los esquemas resultantes R_1, \dots, R_n no contienen una llave de la tabla original, agregar una.

<u>Nombre Comuna</u>	<u>Región</u>	Tasa x m2
A	I	2
A	I	2
B	III	1,1
B	X	1,1
C	V	0,5

<u>RUT</u>	Nombre	Apellido
111111	Claudio	Gonzalez
222222	Maria	Zapata
333333	Carlos	Fernandez
...

RUT \rightarrow Nombre, Apellido

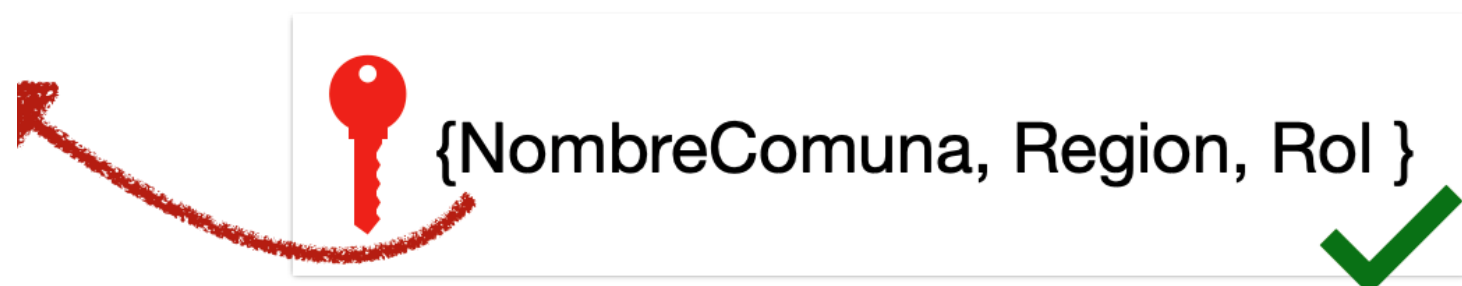
<u>Tasa x m2</u>	<u>M2</u>	Avaluo
34	455	960
35	570	1040
27	895	1790
...

Tasa, m2 \rightarrow Avalúo

NombreComuna, Region \rightarrow
Tasa

<u>Nombre Comuna</u>	<u>Región</u>	RUT	<u>Rol Lote</u>	M2
A	I	111111	34	455

NombreComuna, Region, Rol \rightarrow RUT, M2



Recapitulación

- Partimos desde tablas posiblemente mal diseñadas que generan anomalías
- Agregamos dependencias funcionales
- Intentamos descomponer en BCNF
- Si tenemos problemas con las dependencias utilizar 3NF