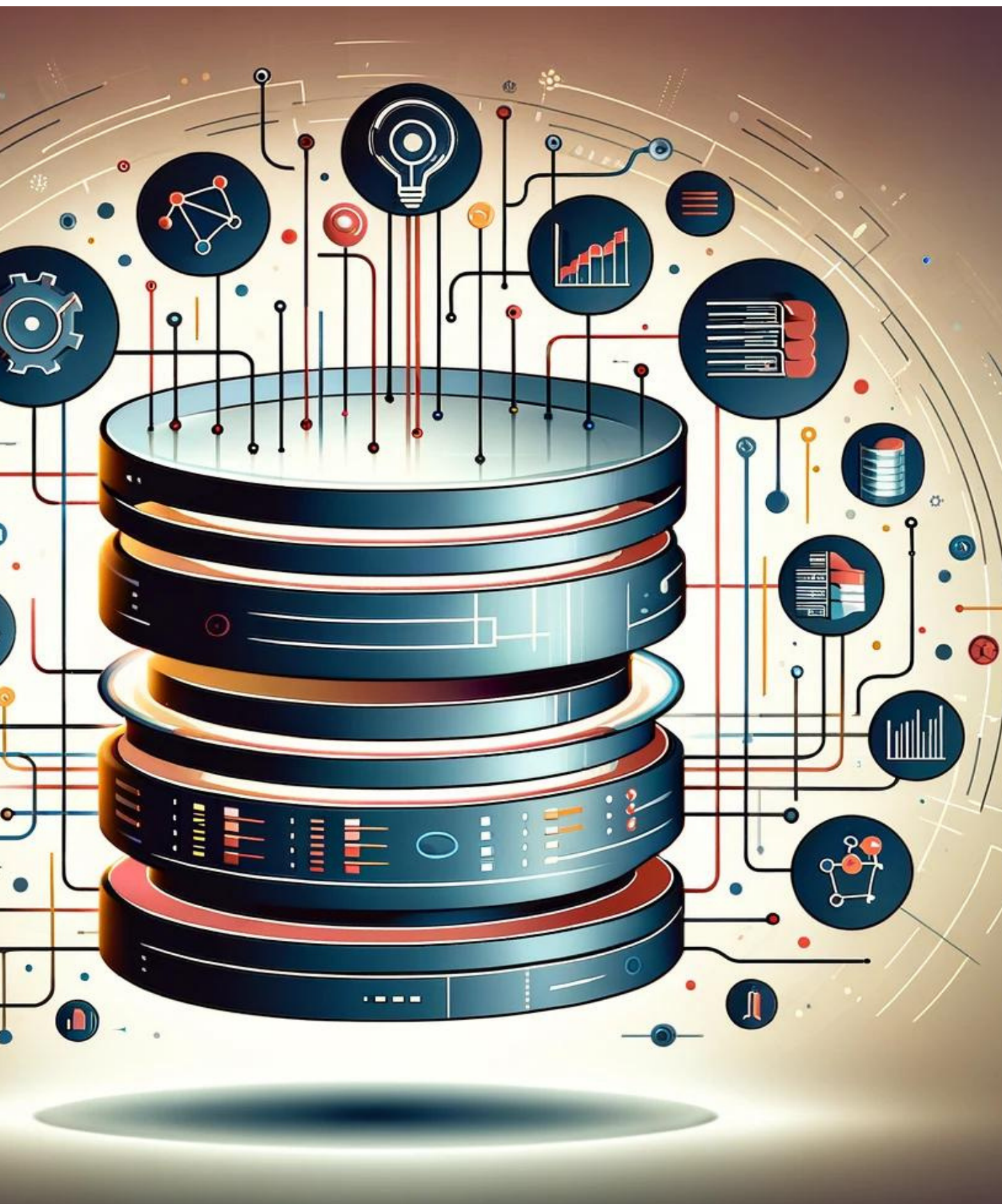




Bases de Datos 1



Alejandra Lliteras
Prof. Titular



Federico Orlando
Prof. Adjunto



TEMAS GENERALES

Bases de Datos 1

1

Modelo de
Datos

2

Teoría de
Diseño de
Bases de
Datos

3

Álgebra
Relacional

4

DBMS
Relacional
MySQL

5

Visualización
de Datos

TEMAS Y SUBTEMAS

Vimos la clase pasada

Teoría de Diseño de Bases de Datos Relacionales

Conceptos Generales:

- Clave de una relación
- Clave candidata
- Superclave
- Axiomas de Armstrong
- Clausura de un conjunto de atributos

Normalización:

- Descomposición
- Pérdida de Información
- Pérdida de Dependencias Funcionales
- Formas Normales
 - Boyce y Codd (BCNF)

TEMAS Y SUBTEMAS

Hoy veremos...

Teoría de Diseño de Bases de Datos Relacionales

Proceso de Normalización a BCNF

Algoritmo para analizar pérdida de dfs

3ra Forma Normal

Proceso de Normalización a 3FN

1ra. Forma Normal

2da. Forma Normal

Dependencias Multivaluadas

4ta Forma Normal

Proceso de Normalización a 4FN

Síntesis del proceso de normalización

Un esquema de relación está en BCNF sí, siempre que una dependencia funcional de la forma $X \rightarrow A$ es válida en R, entonces se cumple que:

- X es superclave de R
o bien
- $X \rightarrow A$ es una dependencia funcional trivial

Forma
Normal
De
Boyce
y
Codd
(FNBC)
(BCNF)


Particionar llevando un esquema a **BCFN**,
asegura que:

- las anomalías dejan de estar (sólo puede quedar redundancia),
- que no se pierda información y,
- en algunos casos, asegura que no se pierdan dependencias funcionales

En general, al
llevar un esquema
a BCNF

Para normalizar un esquema (en principio) y propiciar un buen diseño, vamos a valernos de:

- las dependencias funcionales del esquema
 - las claves candidatas
 - la definición de BCNF
- la descomposición o particionamiento del esquema



Iniciemos con el proceso de normalización de un esquema a partir de un ejemplo

Considerando que el esquema ya se encuentra en Primera Forma Normal



Teoría de Diseño de BBDD Relacionales

Cómo llevar un esquema R a BCNF (cuando se puede)

● Hallar dependencias funcionales y claves candidatas

● 1) Analizar si en el esquema R **existe alguna dependencia funcional** que lleva al esquema a **no cumplir** con la definición de **BCNF**

- **1.1) SI** existe tal dependencia funcional, **particionar** el esquema en dos nuevos esquemas R_i, R_{i+1} , contemplando la dependencia funcional en cuestión.

Analizar las 2 particiones generadas preguntándose:

- **1.1.1) Se pierde información?**
 - 1.1.1.1: NO, entonces sigo a 1.1.2
 - 1.1.1.2: SI. La partición es errónea. Re analizar
- **1.1.2) Se pierden Dependencias funcionales?**
 - 1.1.2.1 NO, entonces sigo a 1.1.3
 - 1.1.2.2 Si. Veremos este caso en breve
- **1.1.3) Determinar en qué forma normal esta R_i, R_{i+1} , si no están en BCNF, reiniciar desde 1, sino pasar a 1.2**
- **1.2) Si **NO** existe, el esquema está en BCNF**

Teoría de Diseño de BDD Relacionales

FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

Donde:

- Hay diversos salones, pero de cada salón se conoce la dirección y la capacidad máxima
- Una fiesta se brinda en un salón y en una fecha determinada. Notar que en un mismo salón se realizan diferentes fiestas en diferentes fechas
- Para cada fiesta se conocen todos los servicios contratados
- Para cada fiesta se saben quiénes son los invitados y donde se debe sentar cada uno de ellos

Hallar la o las claves candidatas y las dependencias funcionales



...trabajando...

Teoría de Diseño de BBDD Relacionales

FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

Donde:

- Hay diversos salones, pero de cada salón se conoce la dirección y la capacidad máxima
- Una fiesta se brinda en un salón y en una fecha determinada. Notar que en un mismo salón se realizan diferentes fiestas en diferentes fechas
- Para cada fiesta se conocen todos los servicios contratados
- Para cada fiesta se saben quiénes son los invitados y donde se debe sentar cada uno de ellos

Clave candidata

(#salon, fecha_fiesta, dni_invitado, servicio_contratado)

Dependencias Funcionales

1. #salon → dirección, capacidad
2. #salon, fecha_fiesta, dni_invitado → mesa_invitado
3. dni_invitado → nombre_invitado

Teoría de Diseño de BBDD Relacionales

Cómo llevar un esquema R a BCNF (cuando se puede)

● Hallar dependencias funcionales y claves candidatas

● **1)** Analizar si en el esquema R **existe alguna dependencia funcional** que lleva al esquema a **no cumplir** con la definición de **BCNF**

- **1.1)** **SI** existe tal dependencia funcional, particionar el esquema en dos nuevos esquemas R_i, R_{i+1} , contemplando la dependencia funcional en cuestión.

Analizar las 2 particiones generadas preguntándose:

- **1.1.1)** Se pierde información?
 - 1.1.1.1: NO, entonces sigo a 1.1.2
 - 1.1.1.2: SI. La partición es errónea. Re analizar
- **1.1.2)** Se pierden Dependencias funcionales?
 - 1.1.2.1 NO, entonces sigo a 1.1.3
 - 1.1.2.2 Si. Veremos este caso en breve
- **1.1.3)** Determinar en qué forma normal esta R_i, R_{i+1} , si no están en BCNF, reiniciar desde 1, sino pasar a 1.2
- **1.2)** Si **NO** existe, el esquema está en BCNF

Teoría de Diseño de BBDD Relacionales

FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

Clave candidata

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

Dependencias Funcionales

1. #salon \rightarrow dirección, capacidad
2. #salon, fecha_fiesta, dni_invitado \rightarrow mesa_invitado
3. dni_invitado \rightarrow nombre_invitado

FIESTAS cumple con la definición de BCNF?

BCNF

Para toda dependencia funcional se cumple que:

X es superclave de R

o bien

X \rightarrow A es una dependencia funcional trivial

Teoría de Diseño de BBDD Relacionales

- FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

Por ejemplo, analizo la dependencia funcional 1:

1. #salon \rightarrow dirección, capacidad

BCNF

Para toda dependencia funcional se cumple que:

X es superclave de R

o bien

X \rightarrow A es una dependencia funcional trivial

{#salon} no es superclave del esquema FIESTAS

FIESTAS no cumple la definición de BCNF

Teoría de Diseño de BBDD Relacionales

Cómo llevar un esquema R a BCNF (cuando se puede)

● Hallar dependencias funcionales y claves candidatas

● **1)** Analizar si en el esquema R **existe alguna dependencia funcional** que lleva al esquema a **no cumplir** con la definición de **BCNF**

- **1.1)** **Si** existe tal dependencia funcional, **particionar** el esquema en dos nuevos esquemas R_i, R_{i+1} , contemplando la dependencia funcional en cuestión.

Analizar las 2 particiones generadas preguntándose:

- 1.1.1) Se pierde información?
 - 1.1.1.1: NO, entonces sigo a 1.1.2
 - 1.1.1.2: Si. La partición es errónea. Re analizar
- 1.1.2) Se pierden Dependencias funcionales?
 - 1.1.2.1 NO, entonces sigo a 1.1.3
 - 1.1.2.2 Si. Veremos este caso en breve
- 1.1.3) Determinar en qué forma normal esta R_i, R_{i+1} , si no están en BCNF, reiniciar desde 1, sino pasar a 1.2

- 1.2) Si **NO** existe, el esquema **está en BCNF**

Teoría de Diseño de BBDD Relacionales

- FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

1. #salon \rightarrow dirección, capacidad

Dado que **FIESTAS NO CUMPLE CON LA DEFINICION DE BCNF**,
descompongo/particiono FIESTAS considerando la dependencia funcional 1.

F1(#salon, dirección, capacidad)

F2 = Fiestas – {dirección, capacidad}

Es decir, F2 tiene los siguientes atributos:

F2 (#salon, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado,
dni_invitado)

Un esquema R donde vale una
dependencia funcional $X \rightarrow Y$ se
descompone como

R1(X, Y)

R2(R-Y)

Teoría de Diseño de BBDD Relacionales

- FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

1. #salon → dirección, capacidad

Dado que FIESTAS NO CUMPLE CON LA DEFINICION DE BCNF, **descompongo/particiono** FIESTAS considerando la df1.

F1(#salon, dirección, capacidad)

F2 (#salon, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

Con el particionamiento propuesto:

¿Se perdió información?

¿Se perdieron dependencias funcionales?

Teoría de Diseño de BBDD Relacionales

Cómo llevar un esquema R a BCNF (cuando se puede)

● Hallar dependencias funcionales y claves candidatas

● 1) Analizar si en el esquema R **existe alguna dependencia funcional** que lleva al esquema a **no cumplir** con la definición de **BCNF**

- 1.1) **Si** existe tal dependencia funcional, particionar el esquema en dos nuevos esquemas R_i , R_{i+1} , contemplando la dependencia funcional en cuestión.

Analizar las 2 particiones generadas preguntándose:

- 1.1.1) **Se pierde información?**

1.1.1.1: NO, entonces sigo a 1.1.2

1.1.1.2: Si. La partición es errónea. Re analizar

- 1.1.2) **Se pierden Dependencias funcionales?**

1.1.2.1 NO, entonces sigo a 1.1.3

1.1.2.2 Si. Veremos este caso en breve

- 1.1.3) Determinar en qué forma normal esta R_i , R_{i+1} , si no están en BCNF, reiniciar desde 1, sino pasar a 1.2

- 1.2) Si **NO** existe, el esquema está en BCNF

Teoría de Diseño de BBDD Relacionales

- FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

1. #salon \rightarrow dirección, capacidad

Dado que FIESTAS NO CUMPLE CON LA DEFINICION DE BCNF, **descompongo/particiono** FIESTAS considerando la df1.

F1(#salon, dirección, capacidad)

F2 (#salon, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

$R1 \cap R2$ clave en el esquema R1
o
 $R1 \cap R2$ clave en el esquema R2

Con el particionamiento propuesto:

¿Se perdió información?

¿Se perdieron dependencias funcionales?

Teoría de Diseño de BBDD Relacionales

Clave candidata: (#salon, fecha_fiesta, dni_invitado, nom_contratante, servicio_contratado)

F1(#salon, direccion, capacidad)

F2(#salon, fecha_fiesta, nom_contratante, cant_invitados, nombre_invitado, cant_mesas, mesa_invitado, servicio_contratado, dir_contratante, dni_invitado)

Con el particionamiento propuesto:

¿Se perdió información?

$F1 \cap F2$ es clave en el esquema {#salon}

Entonces, no se perdió información.

Teoría de Diseño de BBDD Relacionales

- FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

1. #salon → dirección, capacidad

Dado que FIESTAS NO CUMPLE CON LA DEFINICION DE BCNF, **descompongo/particiono** FIESTAS considerando la df1.

F1(#salon, dirección, capacidad)

F2(#salon, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

Verificar que cada una de las dependencias funcionales que valían en el esquema R, sigan valiendo en alguna de las particiones Ri.

Con el particionamiento propuesto:

¿Se perdió información? NO

¿Se perdieron dependencias funcionales?

Teoría de Diseño de BBDD Relacionales

Cómo llevar un esquema R a BCNF (cuando se puede)

● Hallar dependencias funcionales y claves candidatas

● 1) Analizar si en el esquema R **existe alguna dependencia funcional** que lleva al esquema a **no cumplir** con la definición de **BCNF**

- 1.1) **SI** existe tal dependencia funcional, **particionar** el esquema en dos nuevos esquemas R_i, R_{i+1} , contemplando la dependencia funcional en cuestión.

Analizar las 2 particiones generadas preguntándose:

- 1.1.1) **Se pierde información?**
 - 1.1.1.1: NO, entonces sigo a 1.1.2
 - 1.1.1.2: SI. La partición es errónea. Re analizar

- 1.1.2) **Se pierden Dependencias funcionales?**
 - 1.1.2.1 NO, entonces sigo a 1.1.3
 - 1.1.2.2 Si. Veremos este caso en breve

- 1.1.3) Determinar en qué forma normal esta R_i, R_{i+1} , si no están en BCNF, reiniciar desde 1, sino pasar a 1.2

- 1.2) Si **NO** existe, el esquema **está en BCNF**

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

F1(#salon, dirección, capacidad)

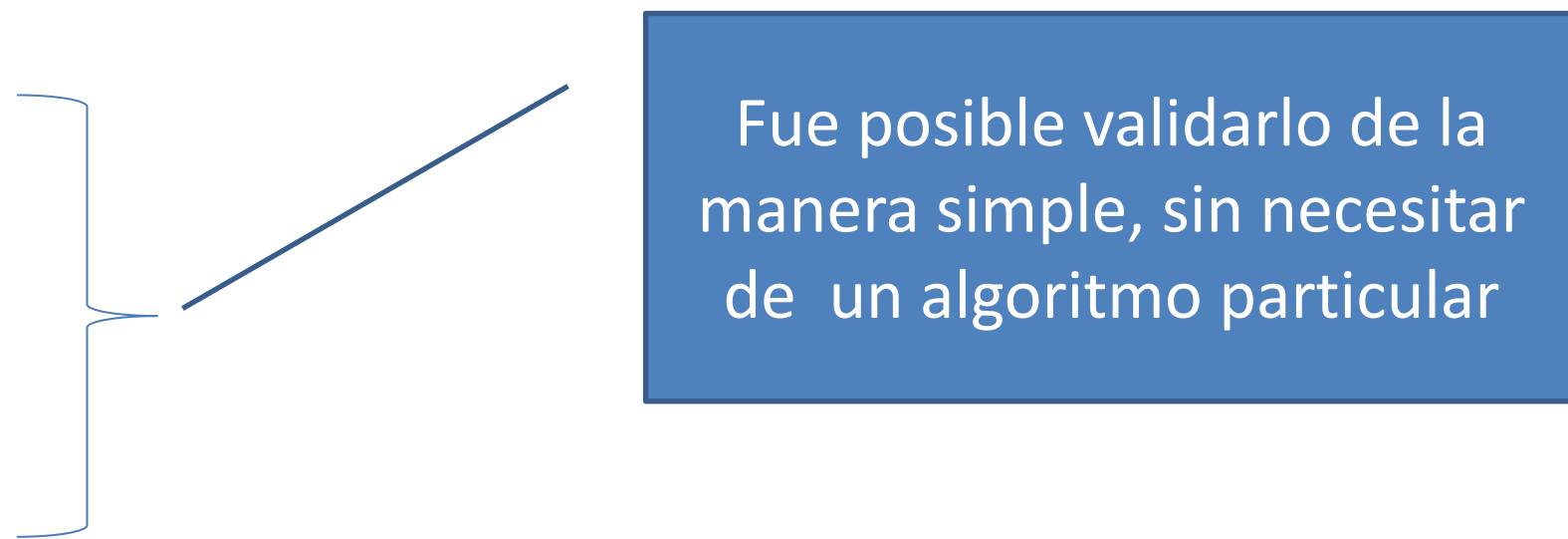
F2(#salon, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado,
dni_invitado)

¿Se perdieron dependencias funcionales?

1. #salon → dirección, capacidad
2. #salon, fecha_fiesta, dni_invitado → mesa_invitado
3. dni_invitado → nombre_invitado

En F1, vale df1

En F2 valen las dfs 2 y 3



Fue posible validarlo de la manera simple, sin necesitar de un algoritmo particular

Entonces, no se perdieron dependencias funcionales.

Teoría de Diseño de BBDD Relacionales

Cómo llevar un esquema R a BCNF (cuando se puede)

● Hallar dependencias funcionales y claves candidatas

● **1)** Analizar si en el esquema R **existe alguna dependencia funcional** que lleva al esquema a **no cumplir** con la definición de **BCNF**

- **1.1)** **Si** existe tal dependencia funcional, particionar el esquema en dos nuevos esquemas R_i, R_{i+1} , contemplando la dependencia funcional en cuestión.

Analizar las 2 particiones generadas preguntándose:

- 1.1.1) Se pierde información?

1.1.1.1: NO, entonces sigo a 1.1.2

1.1.1.2: Si. La partición es errónea. Re analizar

- 1.1.2) Se pierden Dependencias funcionales?

1.1.2.1 NO, entonces sigo a 1.1.3

1.1.2.2 Si. Veremos este caso en breve

- **1.1.3)** Determinar en qué forma normal esta R_i, R_{i+1} , si no están en BCNF, reiniciar desde 1, sino pasar a 1.2

- 1.2) Si **NO** existe, el esquema está en BCNF

Teoría de Diseño de BDD Relacionales

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

F1(#salon, dirección, capacidad)

F2(#salon, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado,
dni_invitado)

¿F1 y F2 cumplen con la definición de BCNF?

BCNF

Para toda dependencia funcional se cumple que:

X es superclave de R

o bien

X→A es una dependencia funcional trivial

En **F1** vale sólo la df1, en F1 se cumple que {#salon} es superclave, entonces

F1 cumple la definición de BCNF

En **F2** vale valen las df2 y df3. En particular, {dni_invitado} no es superclave en F2.

Entonces, **F2 no cumple con la definición de BCNF**, descompongo/particiono F2, considerando la dependencia funcional 3

1. #salon → dirección, capacidad
2. #salon, fecha_fiesta, dni_invitado → mesa_invitado
3. dni_invitado → nombre_invitado

En este caso, debo seguir particionando, ahora con foco en F2



Teoría de Diseño de BDD Relacionales

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

F1(#salon, dirección, capacidad)

F2(#salon, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado,
dni_invitado)

En **F2** vale valen las df2 y df3. En particular, {dni_invitado} no es superclave en F2.

Entonces, **F2 no cumple con la definición de BCNF**, descompongo/particiono F2, considerando la dependencia funcional 3

F3 (dni_invitado, nombre_invitado)

F4(#salon, fecha_fiesta, mesa_invitado, servicio_contratado, dni_invitado)

Con el particionamiento propuesto:

¿Se perdió información?

¿Se perdieron dependencias funcionales?

1. #salon → dirección, capacidad
2. #salon, fecha_fiesta, dni_invitado → mesa_invitado
3. dni_invitado → nombre_invitado

Teoría de Diseño de BDD Relacionales

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

F1(#salon, dirección, capacidad)

F2(#salon, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado,
dni_invitado)

En **F2** vale valen las df2 y df3. En particular, {dni_invitado} no es superclave en F2.

Entonces, **F2 no cumple con la definición de BCNF**, descompongo/particiono F2, considerando la dependencia funcional 3

F3 (dni_invitado, nombre_invitado)

F4(#salon, fecha_fiesta, mesa_invitado, servicio_contratado, dni_invitado)

$R1 \cap R2$ clave en el esquema R1

o

$R1 \cap R2$ clave en el esquema R2

Con el particionamiento propuesto:

¿Se perdió información?

¿Se perdieron dependencias funcionales?

1. #salon \rightarrow dirección, capacidad
2. #salon, fecha_fiesta, dni_invitado \rightarrow mesa_invitado
3. dni_invitado \rightarrow nombre_invitado

Teoría de Diseño de BDD Relacionales

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

F1(#salon, dirección, capacidad)

F2(#salon, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado,
dni_invitado)

En **F2** vale valen las df2 y df3. En particular, {dni_invitado} no es superclave en F2.

Entonces, **F2 no cumple con la definición de BCNF**, descompongo/particiono F2, considerando la dependencia funcional 3

F3 (dni_invitado, nombre_invitado)

F4(#salon, fecha_fiesta, mesa_invitado, servicio_contratado, dni_invitado)

¿Con el particionamiento propuesto se perdió información?

$F3 \cap F4$ es clave en el esquema {dni_invitado}

Entonces, no se perdió información.

Teoría de Diseño de BDD Relacionales

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

F1(#salon, dirección, capacidad)

F2(#salon, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado,
dni_invitado)

En **F2** vale valen las df2 y df3. En particular, {dni_invitado} no es superclave en F2.

Entonces, **F2 no cumple con la definición de BCNF**, descompongo/particiono F2, considerando la dependencia funcional 3

F3 (dni_invitado, nombre_invitado)

F4(#salon, fecha_fiesta, mesa_invitado, servicio_contratado, dni_invitado)

Verificar que cada una de las dependencias funcionales que valían en el esquema R, sigan valiendo en alguna de las particiones Ri.

Con el particionamiento propuesto:

¿Se perdió información?

¿Se perdieron dependencias funcionales?

1. #salon → dirección, capacidad
2. #salon, fecha_fiesta, dni_invitado → mesa_invitado
3. dni_invitado → nombre_invitado

Teoría de Diseño de BDD Relacionales

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

F1(#salon, dirección, capacidad)

F2(#salon, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado,
dni_invitado)

En **F2** vale valen las df2 y df3. En particular, {dni_invitado} no es superclave en F2.

Entonces, **F2 no cumple con la definición de BCNF**, descompongo/particiono F2, considerando la dependencia funcional 3

F3 (dni_invitado, nombre_invitado)

F4(#salon, fecha_fiesta, mesa_invitado, servicio_contratado, dni_invitado)

¿Con el particionamiento propuesto se perdieron dependencias funcionales?

En **F3** vale la dependencia funcional 3. {dni_invitado} es superclave en F3. Entonces, **F3 cumple con la definición de BCNF**.

En **F4** vale la dependencia funcional 2. {#salon, fecha_fiesta, dni_invitado } NO es superclave en F4. Entonces, **F4 NO cumple con la definición de BCNF**.

En este caso, debo seguir particionando, ahora con foco en $F4$



Teoría de Diseño de BDD Relacionales

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

F3 (dni_invitado, nombre_invitado)

F4(#salon, fecha_fiesta, mesa_invitado, servicio_contratado, dni_invitado)

Dado que **F4 NO cumple con la definición de BCNF**, los particiono considerando la dependencia funcional 2

F5 (#salon, fecha_fiesta, dni_invitado, mesa_invitado)

F6(#salon, fecha_fiesta, servicio_contratado, dni_invitado)

$R1 \cap R2$ clave en el esquema $R1$

o

$R1 \cap R2$ clave en el esquema $R2$

Con el particionamiento propuesto:

¿Se perdió información?

¿Se perdieron dependencias funcionales?

1. #salon \rightarrow dirección, capacidad
2. #salon, fecha_fiesta, dni_invitado \rightarrow mesa_invitado
3. dni_invitado \rightarrow nombre_invitado

Teoría de Diseño de BDD Relacionales

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

F3 (dni_invitado, nombre_invitado)

F4(#salon, fecha_fiesta, mesa_invitado, servicio_contratado, dni_invitado)

Dado que **F4 NO cumple con la definición de BCNF**, los particiono considerando la dependencia funcional 2

F5 (#salon, fecha_fiesta, dni_invitado, mesa_invitado)

F6(#salon, fecha_fiesta, servicio_contratado, dni_invitado)

¿Con el particionamiento propuesto se perdió información?

F5 \cap F6 es clave en el esquema {#salon, fecha_fiesta, dni_invitado}

Entonces, no se perdió información.

1. #salon \rightarrow dirección, capacidad
2. #salon, fecha_fiesta, dni_invitado \rightarrow mesa_invitado
3. dni_invitado \rightarrow nombre_invitado

Teoría de Diseño de BDD Relacionales

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

F3 (dni_invitado, nombre_invitado)

F4(#salon, fecha_fiesta, mesa_invitado, servicio_contratado, dni_invitado)

Dado que **F4 NO cumple con la definición de BCNF**, los particiono considerando la dependencia funcional 2

F5 (#salon, fecha_fiesta, dni_invitado, mesa_invitado)

F6(#salon, fecha_fiesta, servicio_contratado, dni_invitado)

Verificar que cada una de las dependencias funcionales que valían en el esquema R, sigan valiendo en alguna de las particiones Ri.

Con el particionamiento propuesto:

¿Se perdió información?

¿Se perdieron dependencias funcionales?

1. #salon → dirección, capacidad
2. #salon, fecha_fiesta, dni_invitado → mesa_invitado
3. dni_invitado → nombre_invitado

Teoría de Diseño de BDD Relacionales

cc: (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

F3 (dni_invitado, nombre_invitado)

F4(#salon, fecha_fiesta, mesa_invitado, servicio_contratado, dni_invitado)

Dado que **F4 NO cumple con la definición de BCNF**, los particiono considerando la dependencia funcional 2

F5 (#salon, fecha_fiesta, dni_invitado, mesa_invitado)

F6(#salon, fecha_fiesta, servicio_contratado, dni_invitado)

¿Con el particionamiento propuesto se perdieron dependencias funcionales?

En **F5** vale la dependencia funcional 2. {#salon, fecha_fiesta, dni_invitado} es superclave en F5.

Entonces, **F5 cumple con la definición de BCNF.**

En **F6** todos los atributos forman parte de la clave, Cualquier dependencia funcional que detecte va a ser trivial. **F6 cumple con la definición de BCNF.**

BCNF

Para toda dependencia funcional se cumple que:

X es superclave de R

o bien

X→A es una dependencia funcional trivial

1. #salon → dirección, capacidad
2. #salon, fecha_fiesta, dni_invitado → mesa_invitado
3. dni_invitado → nombre_invitado

Teoría de Diseño de BBDD Relacionales

Cómo llevar un esquema R a BCNF (cuando se puede)

● Hallar dependencias funcionales y claves candidatas

● **1)** Analizar si en el esquema R **existe alguna dependencia funcional** que lleva al esquema a **no cumplir** con la definición de **BCNF**

- **1.1)** **SI** existe tal dependencia funcional, particionar el esquema en dos nuevos esquemas R_i , R_{i+1} , contemplando la dependencia funcional en cuestión.

Analizar las 2 particiones generadas preguntándose:

- **1.1.1)** Se pierde información?
 - 1.1.1.1: NO, entonces sigo a 1.1.2
 - 1.1.1.2: SI. La partición es errónea. Re analizar
- **1.1.2)** Se pierden Dependencias funcionales?
 - 1.1.2.1 NO, entonces sigo a 1.1.3
 - 1.1.2.2 Si. Veremos este caso en breve
- **1.1.3)** Determinar en qué forma normal esta R_i , R_{i+1} , si no están en BCNF, reiniciar desde 1, sino pasar a 1.2
- **1.2)** Si **NO** existe, el esquema **está en BCNF**

Teoría de Diseño de BDD Relacionales

Las particiones del FIESTAS, que quedaron en BCNF, son:

F1(#salon, direccion, capacidad)

F3(dni invitado, nombre_invitado)

F5(#salon, fecha fiesta, dni invitado, mesa_invitado)

F6(#salon, fecha fiesta, dni invitado, servicio contratado)

El esquema FIESTAS queda así normalizado hasta la Forma Normal de Boyce y Codd

Teoría de Diseño de BBDD Relacionales

Las particiones del FIESTAS, que quedaron en BCNF, son:

F1(#salon, direccion, capacidad)

F3(dni invitado, nombre_invitado)

F5(#salon, fecha_fiesta, dni_invitado)

F6(#salon, fecha_fiesta, dni_invitado, nombre_invitado)

El esquema resultante se encuentra normalizado hasta la Forma Normal de Boyce y Codd



Retomaremos este ejemplo cuando veamos dependencias multivaluadas

Teoría de Diseño de BBDD Relacionales

Cómo llevar un esquema R a BCNF (cuando se puede)

Proceso para
intentar llevar el
esquema a BCNF

- Hallar dependencias funcionales y claves candidatas
- 1) Analizar si en el esquema R **existe alguna dependencia funcional** que lleva al esquema a **no cumplir** con la definición de **BCNF**
 - 1.1) **SI** existe tal dependencia funcional, **particionar** el esquema en dos nuevos esquemas R_i, R_{i+1} , contemplando la dependencia funcional en cuestión.
Analizar las 2 particiones generadas preguntándose:
 - 1.1.1) Se pierde información?
 - 1.1.1.1: NO, entonces sigo a 1.1.2
 - 1.1.1.2: SI. La partición es errónea. Re analizar
 - 1.1.2) Se pierden Dependencias funcionales?
 - 1.1.2.1 NO, entonces sigo a 1.1.3
 - 1.1.2.2 Si. Veremos este caso en breve
 - 1.1.3) Determinar en qué forma normal esta R_i, R_{i+1} , si no están en BCNF, reiniciar desde 1, sino pasar a 1.2
 - 1.2) Si **NO** existe, el esquema **está en BCNF**

Teoría de Diseño de B B D D Relacionales

- ***Hemos visto***

- Como llevar un esquema a BCNF cuando no se pierde información ni dependencias funcionales
- En particular, vimos el caso en el que los atributos de todas las dependencias funcionales quedaron todos incluidos en alguna de las particiones generadas

- **Validación simple**

- ***Resta ver:***

- Qué sucede cuando los atributos de alguna dependencia funcional quedaron distribuidos en más de una partición

- **Validación formal mediante un algoritmo**

- Qué hacer en el caso de que alguna dependencia funcional se pierda y **no se pueda** avanzar con el proceso visto para **llevar el esquema a BCNF**



Ejercicio Práctico

Actividades

- Determinar la o las claves candidatas
- Hallar las dependencias funcionales
- Iniciar el proceso de normalización para llevar el esquema a BCNF

Teoría de Diseño de BDD Relacionales

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Donde:

- Un paciente tiene asignado para cada hospital un número de legajo
- Un legajo en un hospital se asigna a una única persona
- En un hospital trabajan muchos médicos y un médico puede trabajar en diversos hospitales
- Un médico atiende a muchos pacientes
- Cada hospital posee un nombre y el mismo nombre se puede repetir para diferentes hospitales
- Un paciente se atiende en muchos hospitales y de cada hospital que se atiende se registran los médicos que lo atienden



...trabajando...

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajopaciente, dniMedico)

Donde:

- Un paciente tiene asignado para cada hospital un número de legajo
- Un legajo en un hospital se asigna a una única persona
- En un hospital trabajan muchos médicos y un médico puede trabajar en diversos hospitales
- Un médico atiende a muchos pacientes
- Cada hospital posee un nombre y el mismo nombre se puede repetir para diferentes hospitales
- Un paciente se atiende en muchos hospitales y de cada hospital que se atiende se registran los médicos que lo atienden

Dependencias Funcionales

df1) codHospital -> nombreHospital

df2) legajoPaciente, codHospital -> dniPaciente

df3) dniPaciente, codHospital -> legajoPaciente

Claves Candidatas

cc1) {codHospital, legajoPaciente, dniMedico}

cc2) {codHospital, dniPaciente, dniMedico}



Analizo si el esquema cumple BCNF

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Dependencias Funcionales

df1) codHospital \rightarrow nombreHospital

df2) legajoPaciente, codHospital \rightarrow dniPaciente

df3) dniPaciente, codHospital \rightarrow legajoPaciente

Claves Candidatas

cc1) {codHospital, legajoPaciente, dniMedico}

cc2) {codHospital, dniPaciente, dniMedico}

ATENCIONES cumple con la definición de BCNF?

Para toda dependencia funcional se cumple que:

X es superclave de R

o bien

$X \rightarrow A$ es una dependencia funcional trivial

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Dependencias Funcionales

df1) codHospital -> nombreHospital

df2) legajoPaciente, codHospital -> dniPaciente

df3) dniPaciente, codHospital -> legajoPaciente

Claves Candidatas

cc1) {codHospital, legajoPaciente, dniMedico}

cc2) {codHospital, dniPaciente, dniMedico}

Cómo el esquema **ATENCIONES** no cumple con la definición de BCNF, ya que al menos encontramos a la df1 donde {codHospital} no es superclave del esquema **ATENCIONES** y sabemos que se puede particionar para eliminar anomalías, procedemos a particionar **ATENCIONES** contemplando la **df1**

1. codHospital -> nombreHospital

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Dependencias Funcionales

df1) codHospital -> nombreHospital

df2) legajoPaciente, codHospital -> dniPaciente

df3) dniPaciente, codHospital -> legajoPaciente

Claves Candidatas

cc1) {codHospital, legajoPaciente, dniMedico}

cc2) {codHospital, dniPaciente, dniMedico}

Cómo el esquema **ATENCIONES** no cumple con la definición de BCNF, ya que al menos encontramos a la df1 donde {codHospital} no es superclave del esquema **ATENCIONES** y sabemos que se puede particionar para eliminar anomalías, procedemos a particionar **ATENCIONES** contemplando la **df1**.

Al particionar, tenemos:

A1(codHospital, nombreHospital)

A2 (codHospital, dniPaciente, legajoPaciente, dniMedico)

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Dependencias Funcionales

df1) codHospital -> nombreHospital

df2) legajoPaciente, codHospital -> dniPaciente

df3) dniPaciente, codHospital -> legajoPaciente

Claves Candidatas

cc1) {codHospital, legajoPaciente, dniMedico}

cc2) {codHospital, dniPaciente, dniMedico}

A1(codHospital, nombreHospital)

A2 (codHospital, dniPaciente, legajoPaciente, dniMedico)

Con el particionamiento propuesto:

¿Se perdió información?

¿Se perdieron dependencias funcionales?

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Dependencias Funcionales

df1) codHospital -> nombreHospital

df2) legajoPaciente, codHospital -> dniPaciente

df3) dniPaciente, codHospital -> legajoPaciente

Claves Candidatas

cc1) {codHospital, legajoPaciente, dniMedico}

cc2) {codHospital, dniPaciente, dniMedico}

A1(codHospital, nombreHospital)

A2 (codHospital, dniPaciente, legajoPaciente, dniMedico)

Con el particionamiento propuesto:

¿Se perdió información?

$A1 \cap A2$ es clave en el esquema ATENCIONES , {codHospital}

Entonces no se perdió información

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Dependencias Funcionales

df1) codHospital -> nombreHospital

df2) legajoPaciente, codHospital -> dniPaciente

df3) dniPaciente, codHospital -> legajoPaciente

Claves Candidatas

cc1) {codHospital, legajoPaciente, dniMedico}

cc2) {codHospital, dniPaciente, dniMedico}

A1(codHospital, nombreHospital)

A2 (codHospital, dniPaciente, legajoPaciente, dniMedico)

Con el particionamiento propuesto:

¿Se perdieron dependencias funcionales?

En A1, vale df1

En A2 valen las dfs 2 y 3

Entonces, no se perdieron dependencias funcionales.

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Dependencias Funcionales

df1) codHospital -> nombreHospital

df2) legajoPaciente, codHospital -> dniPaciente

df3) dniPaciente, codHospital -> legajoPaciente

Claves Candidatas

cc1) {codHospital, legajoPaciente, dniMedico}

cc2) {codHospital, dniPaciente, dniMedico}

A1(codHospital, nombreHospital)

A2 (codHospital, dniPaciente, legajoPaciente, dniMedico)

¿Ambos esquemas quedaron en BCNF?

- En A1, vale **df1**. Donde {codHospital} es superclave del esquema A1. A1 cumple BCNF.

- En A2 valen las dfs 2 y 3. En particular, existe la df2, donde {legajoPaciente, codHospital} no es superclave de A2. Entonces, podemos afirmar que no cumple BCNF

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Dependencias Funcionales

df1) codHospital -> nombreHospital

df2) legajoPaciente, codHospital -> dniPaciente

df3) dniPaciente, codHospital -> legajoPaciente

Claves Candidatas

cc1) {codHospital, legajoPaciente, dniMedico}

cc2) {codHospital, dniPaciente, dniMedico}

A2 (codHospital, dniPaciente, legajoPaciente, dniMedico)

- En A2 valen las dfs 2 y 3. En particular, existe la **df2**, donde {legajoPaciente, codHospital} no es superclave de A2. Entonces, podemos afirmar que no cumple BCNF.
- Por lo antes mencionado, particionamos A2, para llevar a BCNF el esquema

A3 (codHospital, legajoPaciente, dniPaciente)

A4 (codHospital, legajoPaciente, dniMedico)

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Dependencias Funcionales

df1) codHospital -> nombreHospital

df2) legajoPaciente, codHospital -> dniPaciente

df3) dniPaciente, codHospital -> legajoPaciente

Claves Candidatas

cc1) {codHospital, legajoPaciente, dniMedico}

cc2) {codHospital, dniPaciente, dniMedico}

A3 (codHospital, legajoPaciente, dniPaciente)

A4 (codHospital, legajoPaciente, dniMedico)

¿Ambos esquemas quedaron en BCNF?

- En A3, vale df2 y df3.

- ¿A3 está en BCNF?

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

A3 (codHospital, legajoPaciente, dniPaciente)

A4 (codHospital, legajoPaciente, dniMedico)

- En A3, vale df2 y df3.

df2) legajoPaciente, codHospital -> dniPaciente

df3) dniPaciente, codHospital -> legajoPaciente

- ¿Cuáles son las claves candidatas de A3?

cc1{legajoPaciente, codHospital}

cc2{dniPaciente, codHospital}

A3 cumple con la definición de BCNF?

Para toda dependencia funcional , de la forma $X \rightarrow A$, válida en A3 se cumple que:

X es superclave de R

o bien

$X \rightarrow A$ es una dependencia funcional trivial

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

A3 (codHospital, legajoPaciente, dniPaciente)

A4 (codHospital, legajoPaciente, dniMedico)

- En A3, vale df2 y df3.

df2) legajoPaciente, codHospital -> dniPaciente

df3) dniPaciente, codHospital -> legajoPaciente

- ¿Cuáles son las claves candidatas de A3?

cc1{legajoPaciente, codHospital}

cc2{dniPaciente, codHospital}

A3 cumple con la definición de BCNF?

Si, cumple con la definición, los determinantes de ambas dependencias funcionales son clave (un caso particular de superclave) en A3

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Dependencias Funcionales

df1) codHospital -> nombreHospital

df2) legajoPaciente, codHospital -> dniPaciente

df3) dniPaciente, codHospital -> legajoPaciente

Claves Candidatas

cc1) {codHospital, legajoPaciente, dniMedico}

cc2) {codHospital, dniPaciente, dniMedico}

A3 (codHospital, legajoPaciente, dniPaciente)

A4 (codHospital, legajoPaciente, dniMedico)

A4 cumple con la definición de BCNF?

- En A4, todos los atributos forman parte de la clave del esquema. Cualquier dependencia que se halle va a ser trivial. A4 está en BCNF.

- Ejercicio

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Dependencias Funcionales

df1) codHospital -> nombreHospital

df2) legajoPaciente, codHospital -> dniPaciente

df3) dniPaciente, codHospital -> legajoPaciente

Claves Candidatas

cc1) {codHospital, legajoPaciente, dniMedico}

cc2) {codHospital, dniPaciente, dniMedico}

A3 (codHospital, legajoPaciente, dniPaciente)

A4 (codHospital, legajoPaciente, dniMedico)

¿Qué pasó con la cc2?

- Al decidir particionar A2 contemplando la df2, se define en ese punto del proceso a la cc1 como clave del esquema.



*Cierre del proceso hasta BCNF sobre el
esquema ATENCIONES*

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Esquemas que cumplen BCNF:

A1(codHospital, nombreHospital)

A3 (codHospital, legajoPaciente, dniPaciente)

A4 (codHospital, legajoPaciente, dniMedico)

Clave Primaria

{codHospital, legajoPaciente, dniMedico}

ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Esquemas que cumplen BCNF:

A1(codHospital, nombreHospital)

A3 (codHospital, legajoPaciente, dniPaciente)

A4 (codHospital, legajoPaciente, dniMedico)

Notar que, el último esquema que llevo a BCNF, tiene todos los atributos de una de las claves candidatas

Clave Primaria

{codHospital, legajoPaciente, dniMedico}

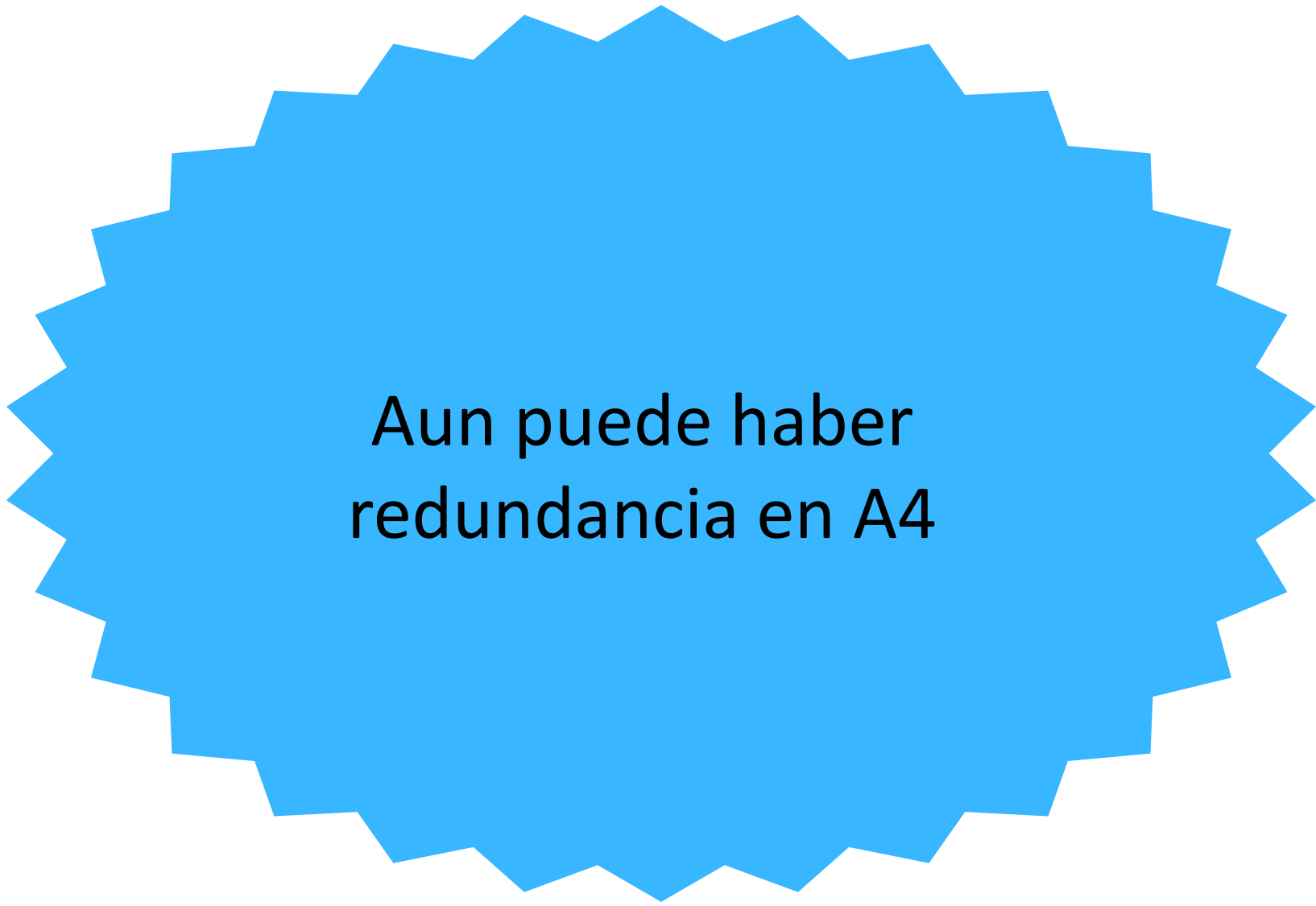
ATENCIONES(codHospital, nombreHospital, dniPaciente, legajoPaciente, dniMedico)

Esquemas que cumplen BCNF:

A1(codHospital, nombreHospital)

A3 (codHospital, legajoPaciente, dniPaciente)

A4 (codHospital, legajoPaciente, dniMedico)



Aun puede haber
redundancia en A4

Clave Primaria

{codHospital, legajoPaciente, dniMedico}

Teoría de diseño de BBDD Relacionales

Análisis de pérdida de dependencias



Teoría de Diseño de B B D D Relacionales

- *Hemos visto*

- Como llevar un esquema a BCNF cuando no se pierde información ni dependencias funcionales
 - En particular, vimos el caso en el que los atributos de todas las dependencias funcionales quedaron todos incluidos en alguna de las particiones generadas

- **Validación simple**

- *Resta ver:*

- Qué sucede cuando los atributos de alguna dependencia funcional quedaron distribuidos en más de una partición

- **Validación formal mediante un algoritmo**

- Qué hacer en el caso de que alguna dependencia funcional se pierda y **no se pueda** avanzar con el proceso visto para **llevar el esquema a BCNF**

Al descomponer, no se debe perder:

- Información
- Dependencias funcionales
 - Validación simple
 - Validación formal mediante un algoritmo

...vimos para la
descomposición
de un esquema

Teoría de Diseño de BBDD Relacionales

Supongamos:

R(a,b,c,d)

Donde vale:

F = { $a \rightarrow b$, $b \rightarrow c$, $c \rightarrow d$, $d \rightarrow a$ }

Y alguien propone el siguiente particionamiento del esquema R

R1(a,b)

R2(b,c)

R3(c,d)

No se pierde información

¿Se pierden dependencias funcionales?

$a \rightarrow b$ vale en **R1**

$b \rightarrow c$ vale en **R2**

$c \rightarrow d$ vale en **R3**

$d \rightarrow a$???????

Res = x

Mientras Res cambia

Para i= 1 _a_ cant_de_ particiones_realizadas

Res = Res $\cup ((\text{Res} \cap \text{Ri})^+ \cap \text{Ri})$

Donde:

- **X** es el determinante de la dependencia funcional que quiero analizar si se perdió.
- **Res** es un temporal donde se van guardando los atributos que se pueden recuperar en cada iteración.
- **Ri** es el conjunto de atributos de la partición representada por Ri
- **$((\text{Res} \cap \text{Ri})^+ \cap \text{Ri})$** , asegura que quedan sólo los atributos que pertenecen a la partición que se está tratando.

**Algoritmo para
analizar la pérdida
de dependencias
funcionales**

Teoría de Diseño de BBDD Relacionales

Supongamos:

R(a,b,c,d)

Donde vale:

F = { $a \rightarrow b$, $b \rightarrow c$, $c \rightarrow d$, $d \rightarrow a$ }

Y alguien propone el siguiente particionamiento del esquema R

R1(a,b)

R2(b,c)

R3(c,d)

No se pierde información

¿Se pierden dependencias funcionales?

$a \rightarrow b$ vale en **R1**

$b \rightarrow c$ vale en **R2**

$c \rightarrow d$ vale en **R3**

$d \rightarrow a$???????

Res = ?

Mientras Res cambia

Para i= 1_a_cant_de_particiones_realizadas

$$\text{Res} = \text{Res} \cup ((\text{Res} \cap R_i)^+ \cap R_i)$$

$R(a,b,c,d)$ $F = \{a \rightarrow b, b \rightarrow c, c \rightarrow d, d \rightarrow a\}$

$R1(a,b)$ $R2(b,c)$ $R3(c,d)$

$d \rightarrow a ?$

X^+

Result:= X

While (hay cambios en result) do

For (cada dependencia funcional $Y \rightarrow Z$ en F)
do

if ($Y \subseteq \text{result}$) then

result := result \cup Z

Res= d

i=1

$$\text{Res} = d \cup ((d \cap \{a,b\})^+ \cap \{a,b\}) = d$$

Paso i=2

$$\text{Res} = d \cup ((d \cap \{b,c\})^+ \cap \{b,c\}) = d$$

i=3

$$\text{Res} = d \cup ((d \cap \{c,d\})^+ \cap \{c,d\})$$

$$\text{Res} = d \cup ((d)^+ \cap \{c,d\})$$

$$\text{Res} = d \cup (\{a,b,c,d\} \cap \{c,d\})$$

$$\text{Res} = d \cup \{c,d\} = \{c,d\}$$

Se itera nuevamente.

i=1

$$\text{Res} = \{c,d\} \cup ((\{c,d\} \cap \{a,b\})^+ \cap \{a,b\})$$

$$= \{c,d\}$$

i=2

$$\text{Res} = \{c,d\} \cup ((\{c,d\} \cap \{b,c\})^+ \cap \{b,c\})$$

$$\text{Res} = \{c,d\} \cup ((c)^+ \cap \{b,c\})$$

$$\text{Res} = \{c,d\} \cup (\{a,b,c,d\} \cap \{b,c\})$$

$$\text{Res} = \{c,d\} \cup \{b,c\} = \{c,d,b\}$$

i=3

$$\text{Res} = \{c,d,b\} \cup ((\{c,d,b\} \cap \{c,d\})^+ \cap \{c,d\}) = \{c,d,b\}$$

Se itera nuevamente.

i=1

$$\text{Res} = \{c,d,b\} \cup ((\{c,d,b\} \cap \{a,b\})^+ \cap \{a,b\}) = \{c,d,b,a\}$$

Se logra incorporar al atributo “a”, a partir del atributo “d”. Entonces no se pierde la dependencia funcional.

Teoría de Diseño de BBDD Relacionales

 Algoritmo para analizar la pérdida de dependencias funcionales

Res = x

Mientras Res cambia

Para i= 1 _a_ cant_de_ particiones_realizadas

Res = Res $\cup ((\text{Res} \cap R_i)^+ \cap R_i)$

SI luego de seguir el algoritmo para detectar pérdida de dependencias, se logra incluir en Res, todos los atributos de la dependencia funcional que sospechaba haber perdido

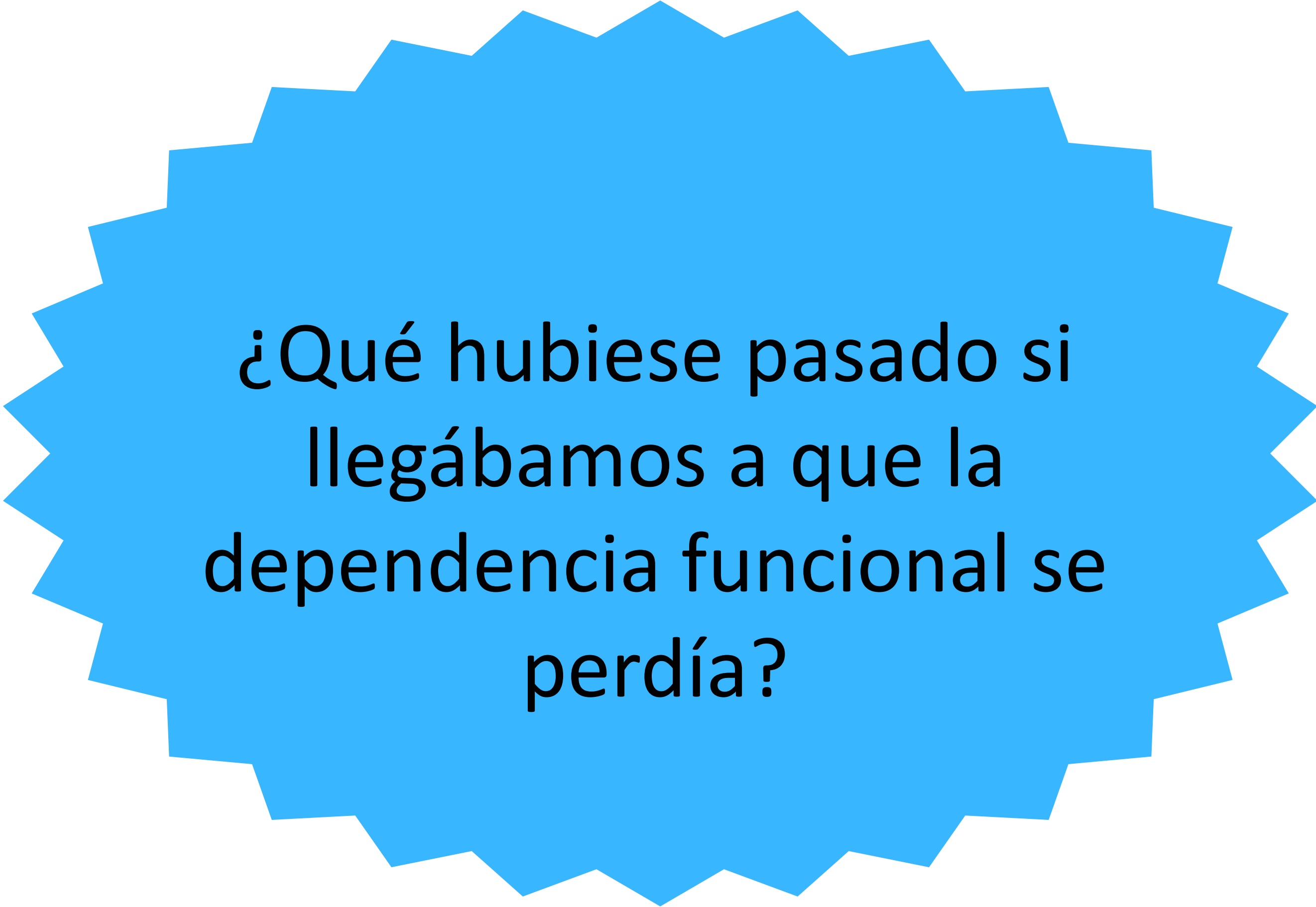


Entonces **NO perdí** la dependencia funcional
Puedo **continuar** con el análisis de **BCNF**

CASO CONTRARIO



PERDÍ la dependencia funcional
NO puedo continuar con el análisis de **BCNF**



¿Qué hubiese pasado si
llegábamos a que la
dependencia funcional se
perdía?



Ejemplo LIBROS

Teoría de Diseño de BBDD Relacionales

LIBROS (titulo, teatro, ciudad)

Donde:

- Un teatro se encuentra en una ciudad
- para cada ciudad en la que se presenta un título de libro, se conoce el teatro.
- Un título se presenta en diferentes ciudades

Valen las siguientes dependencias funcionales

df1) teatro-> ciudad

df2) titulo, ciudad->teatro

Claves candidatas:

cc1: {titulo, ciudad}

cc2: {teatro, titulo}

Teoría de Diseño de BDD Relacionales

LIBROS (titulo, teatro, ciudad)

Valen las siguientes dependencias funcionales

df1) teatro \rightarrow ciudad

df2) titulo, ciudad \rightarrow teatro

Claves candidatas:

cc1: {titulo, ciudad}

cc2: {teatro, titulo}

¿LIBROS cumple con la definición de BCNF?

Para toda dependencia funcional se cumple que:

X es superclave de R

o bien

X \rightarrow A es una dependencia funcional trivial

Teoría de Diseño de BDD Relacionales

LIBROS (titulo, teatro, ciudad)

Valen las siguientes dependencias funcionales

df1) teatro \rightarrow ciudad

df2) titulo, ciudad \rightarrow teatro

Claves candidatas:

cc1: {titulo, ciudad}

cc2: {teatro, titulo}

LIBROS no cumple con la definición de BCNF

Existe la df1, tal que {teatro} no es superclave del esquema.

LIBROS (titulo, teatro, ciudad)

Valen las siguientes dependencias funcionales

df1) teatro-> ciudad

df2) titulo, ciudad->teatro

Claves candidatas:

cc1: {titulo, ciudad}

cc2: {teatro, titulo}

Divido el esquema LIBROS contemplando la df1 ya que {teatro} no es superclave de dicho esquema.

L1(teatro, ciudad)

L2(teatro, titulo)

LIBROS (titulo, teatro, ciudad)

Valen las siguientes dependencias funcionales

df1) teatro-> ciudad

df2) titulo, ciudad->teatro

Claves candidatas:

cc1: {titulo, ciudad}

cc2: {teatro, titulo}

Divido el esquema LIBROS contemplando la df1 ya que {teatro} no es superclave de dicho esquema.

L1(teatro, ciudad)

L2(teatro, titulo)

Con el particionamiento propuesto:

¿Se perdió información?

$L1 \cap L2$ es clave en el esquema $L1$ {teatro}

¿Se perdieron dependencias funcionales?

LIBROS (titulo, teatro, ciudad)

Valen las siguientes dependencias funcionales

df1) teatro-> ciudad

df2) titulo, ciudad->teatro

Claves candidatas:

cc1: {titulo, ciudad}

cc2: {teatro, titulo}

Divido el esquema LIBROS contemplando la df1 ya que {teatro} no es superclave de dicho esquema.

L1(teatro, ciudad)

L2(teatro, titulo)

Con el particionamiento propuesto:

¿Se perdieron dependencias funcionales?

En L1, vale la df1

¿Qué pasó con la df2 (titulo, ciudad->teatro)?

LIBROS (titulo, teatro, ciudad)

Valen las siguientes dependencias funcionales

df1) teatro-> ciudad

df2) titulo, ciudad->teatro

Claves candidatas:

c

En este caso debemos aplicar el algoritmo para determinar pérdida de dependencias funcionales

contemplando la df1 ya que {teatro} no es una.

Con el particionamiento propuesto:

¿Se perdieron dependencias funcionales?

En L1, vale la df1

¿Qué pasó con la df2 (titulo, ciudad->teatro)?

Res = ?

Mientras Res cambia

Para i= 1_a_cant_de_particiones_realizadas

Res = Res \cup ((Res \cap Ri)⁺ \cap Ri)

LIBROS (titulo, teatro, ciudad)

F= {teatro-> ciudad

titulo, ciudad->teatro}

L1(teatro, ciudad)

L2(teatro, titulo)

X⁺

Result:= X

While (hay cambios en result) do

For (cada dependencia funcional Y->Z en F) do

if (Y \subseteq result) then

result := result \cup Z

Res= (titulo, ciudad)

i=1

Res= (titulo, ciudad) \cup (((titulo, ciudad) \cap {teatro,ciudad})⁺ \cap {teatro,ciudad}) =

(titulo, ciudad) \cup ({ciudad})⁺ \cap {teatro,ciudad} =

(titulo, ciudad) \cup ({ \emptyset } \cap {teatro,ciudad})= (titulo, ciudad) \cup { \emptyset } = (titulo, ciudad)

i=2

Res= (titulo, ciudad) \cup (((titulo, ciudad) \cap {teatro,titulo})⁺ \cap {teatro,titulo}) =

(titulo, ciudad) \cup ({titulo})⁺ \cap {teatro,titulo} =

(titulo, ciudad) \cup ({ \emptyset } \cap {teatro,titulo})= (titulo, ciudad) \cup { \emptyset } = (titulo, ciudad)

Terminamos de recorrer todas las particiones de LIBROS y res no cambi3

La df: titulo, ciudad->teatro se perdi3 en el particionamiento!!

LIBROS (titulo, teatro, ciudad)

Valen las siguientes dependencias funcionales

df1) teatro-> ciudad

df2) titulo, ciudad->teatro

Claves candidatas:

ci

Se pierde la dependencia funcional con este particionamiento

contemplando la df1 ya que {teatro} no es una.

Con el particionamiento propuesto:

¿Se perdieron dependencias funcionales?

En L1, vale la df1

¿Qué pasó con la df2 (titulo, ciudad->teatro)?

LIBROS (titulo, teatro, ciudad)

Valen las siguientes dependencias funcionales

df1) teatro-> ciudad

df2) titulo, ciudad->teatro

Claves candidatas:

ci

Se pierde la dependencia funcional con este particionamiento

contemplando la dependencia.

Con el

¿Se p

En L1, v

¿Qué pasó con la dependencia (titulo, ciudad->teatro)?

NO puedo seguir tratando de llevar a BCNF

Teoría de Diseño de BBDD Relacionales

Cómo llevar un esquema R a BCNF (cuando se puede)

NO SE PUEDE

- Hallar dependencias funcionales y claves candidatas

- 1) Analizar si en el esquema R **existe alguna dependencia funcional** que lleva al esquema a **no cumplir** con la definición de **BCNF**

- 1.1) **SI** existe tal dependencia funcional, particionar el esquema en dos nuevos esquemas R_i, R_{i+1} , contemplando la dependencia funcional en cuestión.

Analizar las 2 particiones generadas preguntándose:

- 1.1.1) Se pierde información?
 - 1.1.1.1: NO, entonces sigo a 1.1.2
 - 1.1.1.2: SI. La partición es errónea. Re analizar

- 1.1.2) Se pierden Dependencias funcionales?

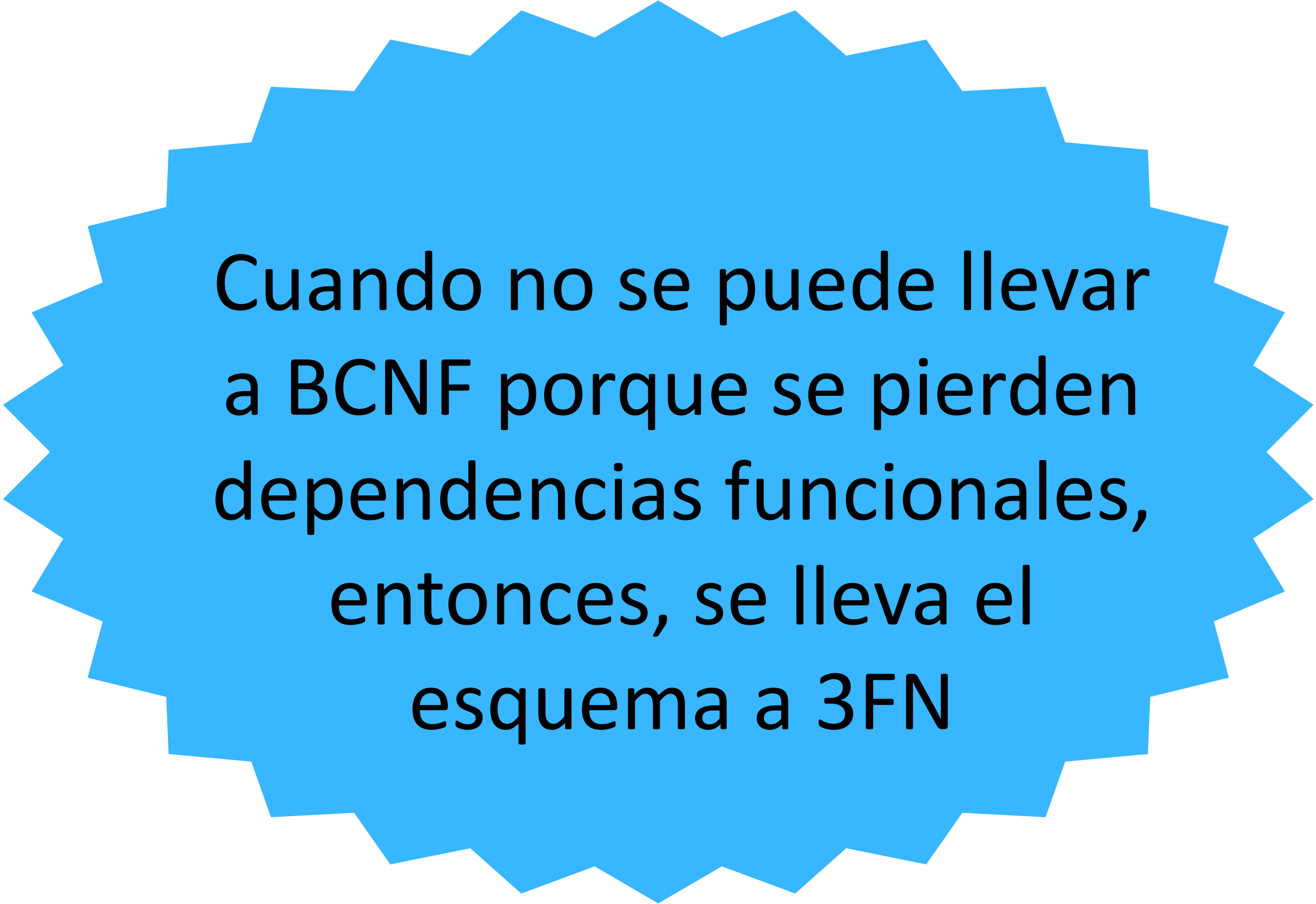
1.1.2.1 NO, entonces sigo a 1.1.3

1.1.2.2 Si. Veremos este caso en breve

SE PERDIERON DEPENDENCIAS FUNCIONALES

- 1.1.3) Determinar en qué forma normal esta R_i, R_{i+1} , si no están en BCNF, reiniciar desde 1, sino pasar a 1.2

- 1.2) Si **NO** existe, el esquema está en BCNF



Cuando no se puede llevar
a BCNF porque se pierden
dependencias funcionales,
entonces, se lleva el
esquema a 3FN

En este escenario, la **3FN**, lo que asegura es que:

- no se pierde información,
- no se pierdan dependencias funcionales,
- pero no siempre se quitan las anomalías.

Cuando no se
puede llevar a
BCNF porque se
pierden
dependencias
funcionales,
entonces se deja
en Tercera Forma
Normal (3FN)


Tercera Forma Normal (3FN)

Un esquema de relación R está en 3FN si para toda dependencia de la forma $X \rightarrow A$, se cumple que:

- $X \rightarrow A$ es trivial
O bien,
- X es superclave
O bien
- A es primo



Atributo primo:
atributo que forma parte de alguna clave candidata



¿Cómo se lleva un esquema a 3FN cuándo no se puede llevar a BCNF porque se pierden dependencias funcionales?

- Se construye una tabla por cada dependencia funcional
- Si la clave de la tabla original no está incluida en ninguna de las tablas del punto anterior, se construye una tabla con la clave

¿Cómo se lleva un
esquema a 3FN cuándo
no se puede llevar a
BCNF porque se
pierden dependencias
funcionales?
*(ya se tienen las dependencias
funcionales y las claves
candidatas)*

LIBROS (titulo, teatro, ciudad)

Valen las siguientes dependencias funcionales

df1) teatro-> ciudad

df2) titulo, ciudad->teatro

Claves candidatas:

cc1: {titulo, ciudad}

cc2: {teatro, titulo}

Como no es posible llevar el esquema a BCNF sin perder dependencias funcionales, entonces, aplico el proceso para dejar el esquema en 3FN.

A (teatro, ciudad)

B (titulo,ciudad, teatro)

(en este caso, como la clave quedo en una de las particiones, no se agrega una nueva partición con ella)

Las particiones A y B están en 3FN

En síntesis



Llamamos **normalizar** hasta BCNF o 3FN, al **proceso** que involucra, hasta ahora, los siguientes pasos:

- Encontrar las dependencias funcionales y las claves candidatas
- Llevar a BCNF aplicando el proceso de descomposición/particionamiento sin pérdida de información
 - Comprobar que no se pierden dependencias funcionales en la descomposición
 - »No, continuo el proceso a BCNF
 - »Si, se lleva el esquema correspondiente a 3NF

En síntesis

Otras Formas Normales

- **1FN**

Los atributos de la relación son simples y atómicos

- **2FN:**

Un esquema de relación R está en 2FN si para toda dependencia de la forma $X \rightarrow A$, se cumple que:

A depende de manera total de la clave



Ejemplo para 2FN

Teoría de Diseño de BBDD Relacionales

- **2FN:**

Un esquema de relación R está en 2FN si para toda dependencia de la forma $X \rightarrow A$, se cumple que: A depende de manera total de la clave

Empleados (#emp, #area, nombre, nombre_area, años)

Donde

- “nombre” es el nombre de un empleado.
 - “años” es la cantidad de años que un empleado trabajó en un área determinada.
 - “nombre_area” es el nombre de un área de trabajo, puede repetirse para distintos #area.
-
- Clave candidata: (#empleado, #area)
 - Dependencias funcionales:
 - 1) #empleado \rightarrow nombre
 - 2) #area \rightarrow nombre_area
 - 3) #empleado, #area \rightarrow años

Teoría de Diseño de BBDD Relacionales

- **2FN:**
Un esquema de relación R está en 2FN si para toda dependencia de la forma $X \rightarrow A$, se cumple que: A depende de manera total de la clave

Empleados (#emp, #area, nombre, nombre_area, años)

- Clave candidata: (#empleado, #área)
- Dependencias funcionales:
 - df1) #empleado \rightarrow nombre
 - df2) #area \rightarrow nombre_area
 - df3) #empleado, #area \rightarrow años

La relación **Empleados no se encuentra en 2FN** porque existen atributos que no dependen funcionalmente de toda la clave de **Empleados** .

Por ejemplo, existe la DF1 en donde ***nombre*** no depende funcionalmente de toda la clave

- Si un esquema está en BCNF, se puede asegurar que además cumple: 3FN, 2FN y 1FN

- Si un esquema está en 3FN, se puede asegurar que además cumple: 2FN y 1FN

Y por ello, no analizaremos 1Fn y 2FN, por el proceso de normalización que se lleva a cabo en la materia (siguiendo al autor: J.D.Ullman).


En general

En síntesis



Teoría de Diseño de B B D D Relacionales

- Particionar llevando un esquema a **BCFN**, asegura que:
 - las anomalías dejan de estar (sólo puede quedar redundancia),
 - que no se pierda información y,
 - en algunos casos, asegura que no se pierdan dependencias funcionales
- Cuando no se puede llevar a BCNF porque se pierden dependencias funcionales, entonces, se lleva el esquema a **3FN**, lo que asegura que:
 - no se pierde información,
 - no se pierdan dependencias funcionales,
 - pero no siempre se quitan las anomalías.



Dejar un esquema en la 3FN o
en BCNF no asegura eliminar
la redundancia

**Hasta acá analizamos la posibilidad de
NORMALIZAR un esquema considerando
las DEPENDENCIAS FUNCIONALES**





*Habíamos visto el proceso hasta BCNF para el
esquema FIESTAS*

Teoría de Diseño de BBDD Relacionales

FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

Donde:

- Hay diversos salones, pero de cada salón se conoce la dirección y la capacidad máxima
- Una fiesta se brinda en un salón y en una fecha determinada. Notar que en un mismo salón se realizan diferentes fiestas en diferentes fechas
- Para cada fiesta se conocen todos los servicios contratados
- Para cada fiesta se saben quiénes son los invitados y donde se debe sentar cada uno de ellos

Teoría de Diseño de BDD Relacionales

FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

Donde:

- Hay diversos salones, pero de cada salón se conoce la dirección y la capacidad máxima
- Una fiesta se brinda en un salón y en una fecha determinada. Notar que en un mismo salón se realizan diferentes fiestas en diferentes fechas
- Para cada fiesta se conocen todos los servicios contratados
- Para cada fiesta se saben quiénes son los invitados y donde se debe sentar cada uno de ellos

Dependencias Funcionales

df1) #salon → dirección, capacidad

df2) #salon, fecha_fiesta, dni_invitado → mesa_invitado

df3) dni_invitado → nombre_invitado

Clave Candidata

(#salon, fecha_fiesta,
dni_invitado, servicio_contratado)

Particiones en BCNF

F1(#salon, direccion, capacidad)

F3(dni invitado, nombre_invitado)

F5(#salon, fecha fiesta, dni invitado, mesa_invitado)

F6(#salon, fecha fiesta, dni invitado, servicio contratado)

Teoría de Diseño de BBDD Relacionales

Las particiones del FIESTAS, que quedaron en BCNF, son:

F1(#salon, direccion, capacidad)

F3(dni invitado, nombre_invitado)

F5(#salon, fecha_fiesta, dni_invitado)

F6(#salon, fecha_fiesta, dni_invitado, nombre_invitado)

El esquema resultante se encuentra normalizado hasta la Forma Normal de Boyce y Codd



Retomaremos este ejemplo cuando veamos dependencias multivaluadas

FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

Donde:

- Hay diversos salones, pero de cada salón se conoce la dirección y la capacidad máxima
- Una fiesta se brinda en un salón y en una fecha determinada. Notar que en un mismo salón se realizan diferentes fiestas en diferentes fechas
- Para cada fiesta se conocen todos los servicios contratados
- Para cada fiesta se saben quiénes son los invitados y donde se debe sentar cada uno de ellos

Teoría de Diseño de BDD Relacionales

¿Cómo podría ser una instancia de F6?

F6(#salon, fecha_fiesta, servicio_contratado, dni_invitado)

#salon	fecha_fiesta	serv_contratado	dni_invitado
#1	10/03/12	empanadas	11111111
#1	10/03/12	empanadas	11222222
#1	10/03/12	pizza	11111111
#1	10/03/12	pizza	11222222
#1	11/03/12	Mesa de quesos	11333333
#1	11/03/12	calentitos	11333333
...

Se puede decir que:

$$X \twoheadrightarrow Y$$

si:

- dado un valor de **X**, hay un conjunto de valores de Y asociados y
- este conjunto de valores de Y **NO** está relacionado (ni funcional ni multifuncionalmente) con los valores de **R - X - Y** (donde **R** es el esquema),

es decir,

Y es independiente de los atributos de **R-X-Y**.

Dependencia
Multivaluada
(DM)

Sea R un esquema de relación

Una **dependencia multivaluada** de la forma

$$X \twoheadrightarrow Y$$

que vale en R es **trivial** si:

- el conjunto de atributos X,Y (X unión Y) son todos los atributos del esquema R

Dependencia
Multivaluada
Trivial
(*DM Trivial*)

Sea R un esquema de relación

- Es posible definir una **dependencia multivaluada** de la forma

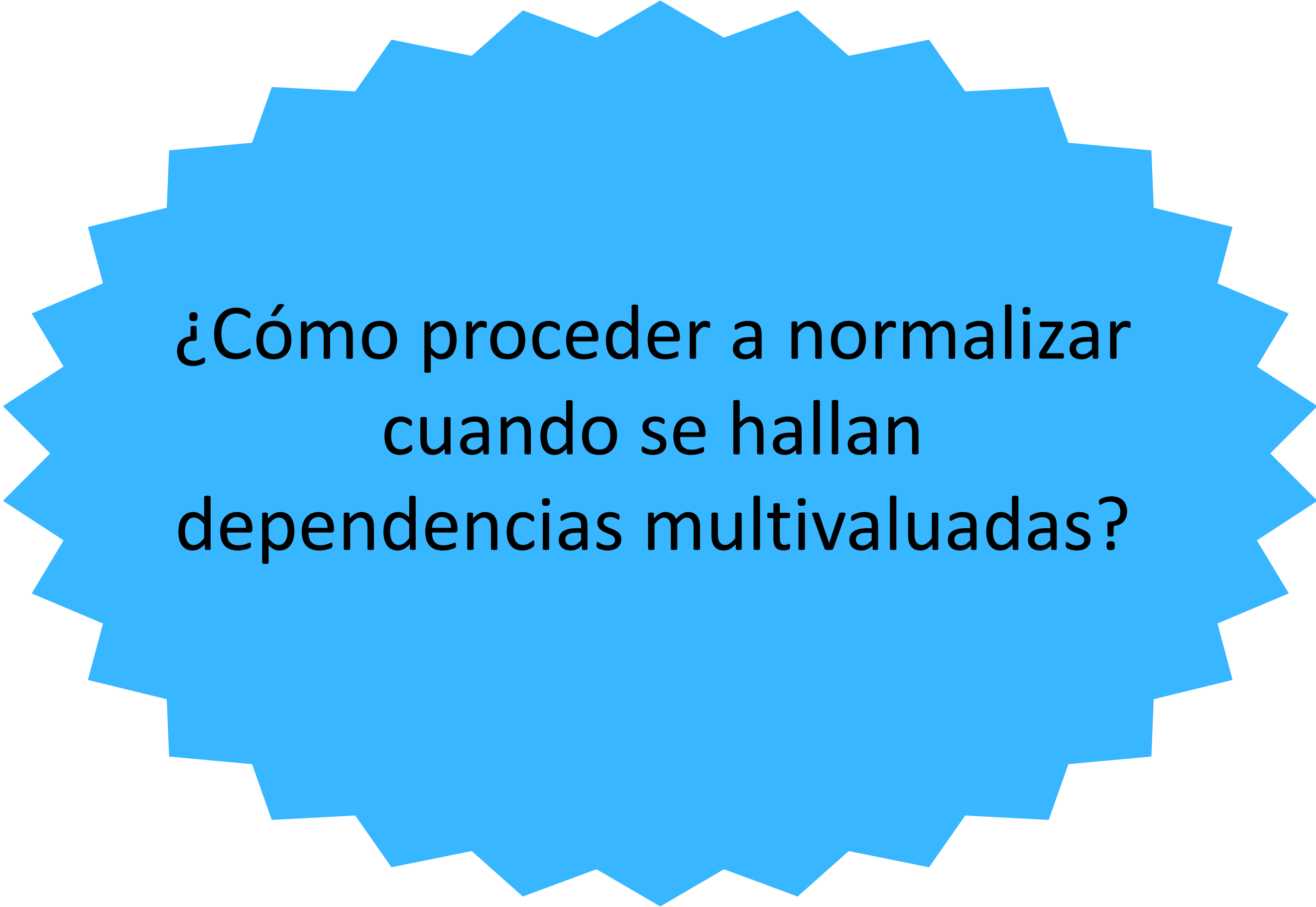
$$\emptyset \twoheadrightarrow Y$$

Y se lee:

vacío multidetermina a Y

Dependencia
Multivaluada
Caso Especial

\emptyset (vacío)



¿Cómo proceder a normalizar
cuando se hallan
dependencias multivaluadas?

- Un esquema R está en 4NF con respecto a un conjunto de dependencias multivaluadas D, si

\forall dependencia multivaluada (DM) de la forma

$$X \twoheadrightarrow Y$$

se cumple que:

$X \twoheadrightarrow Y$ es una DM trivial



4 F N

En otras palabras:

Un esquema R está en 4FN cuando:

- No tiene dependencias multivaluadas o bien,
- Las dependencias multivaluadas que en él valen, son triviales.

4 F N



*Volvemos al esquema FLESTAS que
normalizamos hasta BCNF*

Teoría de Diseño de BDD Relacionales

FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

Donde:

- Hay diversos salones, pero de cada salón se conoce la dirección y la capacidad máxima
- Una fiesta se brinda en un salón y en una fecha determinada. Notar que en un mismo salón se realizan diferentes fiestas en diferentes fechas
- Para cada fiesta se conocen todos los servicios contratados
- Para cada fiesta se saben quiénes son los invitados y donde se debe sentar cada uno de ellos

Dependencias Funcionales

df1) #salon → dirección, capacidad

df2) #salon, fecha_fiesta, dni_invitado → mesa_invitado

df3) dni_invitado → nombre_invitado

Clave Candidata

(#salon, fecha_fiesta,
dni_invitado, servicio_contratado)

Particiones en BCNF

F1(#salon, direccion, capacidad)

F3(dni invitado, nombre_invitado)

F5(#salon, fecha fiesta, dni invitado, mesa_invitado)

F6(#salon, fecha fiesta, dni invitado, servicio contratado)

Teoría de Diseño de BBDD Relacionales

FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

Donde:

- Hay diversos salones, pero de cada salón se conoce la dirección y la capacidad máxima
- Una fiesta se brinda en un salón y en una fecha determinada. Notar que en un mismo salón se realizan diferentes fiestas en diferentes fechas
- Para cada fiesta se conocen todos los servicios contratados
- Para cada fiesta se saben quiénes son los invitados y donde se debe sentar cada uno de ellos

Teoría de Diseño de BBDD Relacionales

F6(#salon, fecha fiesta, servicio contratado, dni invitado)

¿Las siguientes propuestas son válidas de acuerdo a la semántica de los atributos?

#salon -->> DNI_Invitado ? **X** El invitado depende de la fecha de la fiesta
#salon, fecha_fiesta -->> DNI_invitado ? **✓**

#salón	Fecha_fiesta	Serv_contr	Dni_invitado
#1	10/03/12	empanadas	11111111
#1	10/03/12	empanadas	11222222
#1	10/03/12	pizza	11111111
#1	10/03/12	pizza	11222222
#1	11/03/12	Mesa de quesos	11333333
#1	11/03/12	calentitos	11333333
...

X -->> Y si dado un valor de **X**, hay un conjunto de valores de Y asociados y este conjunto de valores de Y **NO** está relacionado (ni funcional ni multifuncionalmente)
con los valores de **R - X -Y** (donde **R** es el esquema), es decir, **Y es independiente** de los atributos de **R-X-Y**.

FIESTAS (#salon, dirección, capacidad, fecha_fiesta, nombre_invitado, mesa_invitado, servicio_contratado, dni_invitado)

Donde:

- Hay diversos salones, pero de cada salón se conoce la dirección y la capacidad máxima
- Una fiesta se brinda en un salón y en una fecha determinada. Notar que en un mismo salón se realizan diferentes fiestas en diferentes fechas
- Para cada fiesta se conocen todos los servicios contratados
- Para cada fiesta se saben quiénes son los invitados y donde se debe sentar cada uno de ellos

F6(#salon, fecha_fiesta, servicio_contratado, dni_invitado)



¿Qué dependencias multivaluadas valen en F6?

DM1) #salon, fecha_fiesta -->> DNI_invitado

DM2) #salon, fecha_fiesta -->> servicio_contratado

**Análisis para llevar a 4FN un esquema
que ya se encuentra en BCNF**





*Llevaremos a 4FN
a F6 que ya se encuentra en BCNF*

Teoría de Diseño de B B D D Relacionales

F6(#salon, fecha_fiesta,dni invitado, servicio contratado)

Dependencias Multivaluadas válidas sobre F6:

DM1) #salon, fecha_fiesta ->> dni_invitado

DM2) #salon, fecha_fiesta ->> servicio_contratado

El esquema F6 **no está en 4NF porque** existen dependencias multivaluadas 1-2 que no son triviales en F6.

Entonces se particiona F6 utilizando una de las dependencias multivaluadas, por ejemplo, DM1

Teoría de Diseño de BBDD Relacionales

F6(#salon, fecha_fiesta,dni invitado, servicio contratado)

Dependencias Multivaluadas válidas sobre F6:

DM1) #salon, fecha_fiesta ->> dni_invitado

DM2) #salon, fecha_fiesta ->> servicio_contratado

F7(#salon, fechaFiesta, dniInvitado)

F8(#salon, fechaFiesta, servicio contratado)

La partición F7 **está en 4NF** ya que sólo vale la dependencia multivaluada 1 que es trivial en ella.

La partición F8 **está en 4NF** porque sólo vale la dependencia multivaluada 2 que es trivial en ella.

Teoría de Diseño de BBDD Relacionales

F6(#salon, fecha_fiesta,dni invitado, servicio contratado)

Dependencias Multivaluadas válidas sobre F6:

DM1) #salon, fecha_fiesta ->> dni_invitado

DM2) #salon, fecha_fiesta ->> servicio_contratado

F7(#salon, fechaFiesta, dniInvitado)


F8(#salon, fechaFiesta, servicio contratado)



Todos los atributos forman parte de la clave

La partición F7 **está en 4NF** ya que sólo vale la dependencia multivaluada 1 que es trivial en ella.


La partición F8 **está en 4NF** porque sólo vale la dependencia multivaluada 2 que es trivial en ella.




Particiones de FLESTA que quedaron en 4FN

Teoría de Diseño de BDD Relacionales

- Esquemas en 4FN
 - F1(#salon, dirección, capacidad)
 - F3 (dni invitado, nombre_invitado)
 - F5 (#salon, fecha fiesta, dni invitado, mesa invitado)
 - F7(#salon, fechaFiesta, dniInvitado)
 - F8(#salon, fechaFiesta, servicio contratado)
- CP (#salon, fecha_fiesta, dni_invitado, servicio_contratado)



¿Porqué
F1, F3 y F5
cumplen con la 4FN?



*F1, F3 y F5
cumplen con la 4FN porque
no tienen DMs*

Teoría de Diseño de BDD Relacionales

- Esquemas en 4FN que no son proyección de otras particiones
 - F1(#salon, dirección, capacidad)
 - F3 (dni invitado, nombre_invitado)
 - F5 (#salon, fecha fiesta, dni invitado, mesa_invitado)
 - F8(#salon, fechaFiesta, servicio contratado)
- CP (#salon, fecha_fiesta, dni_invitado, servicio_contratado)

En síntesis...

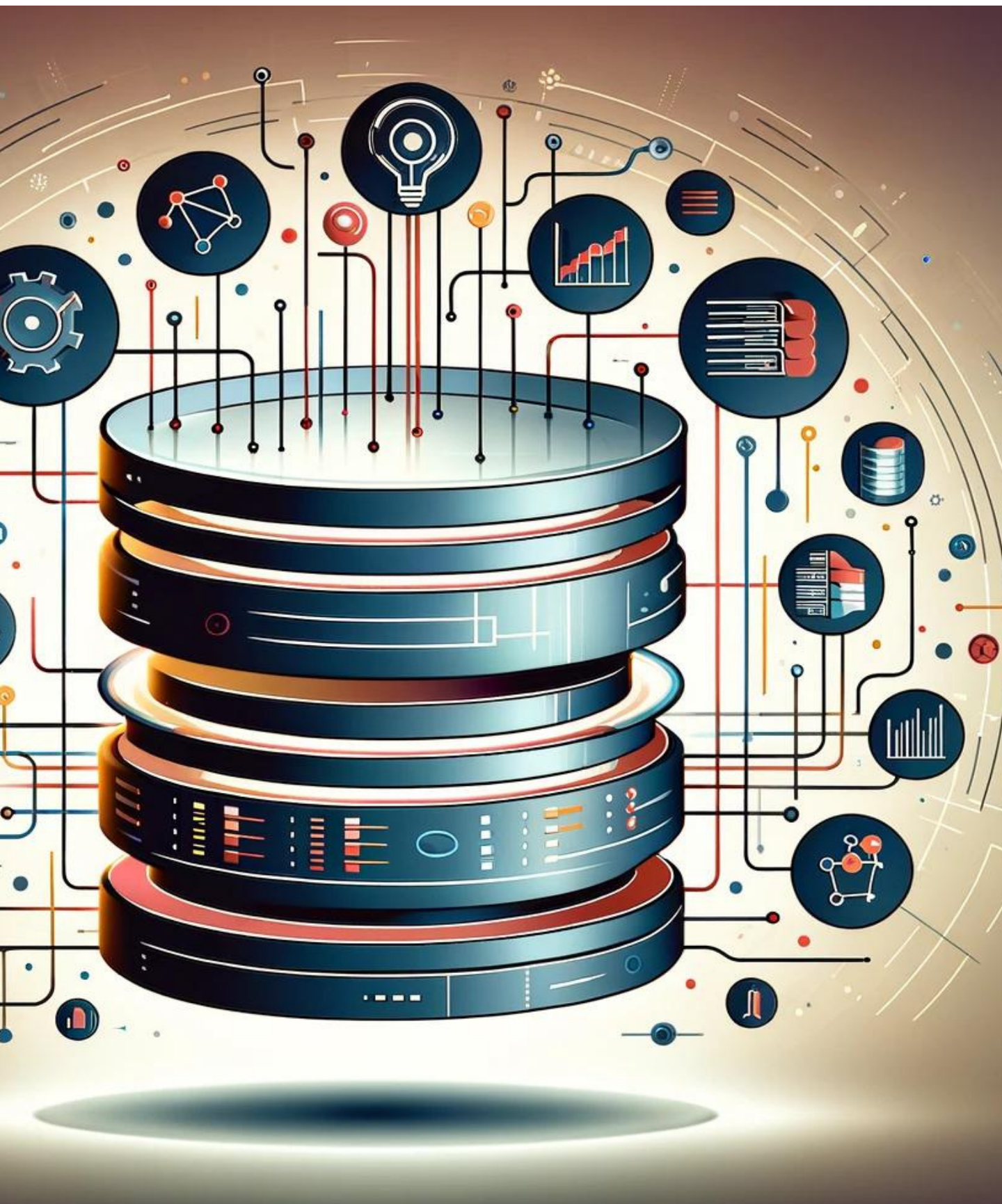


Resumiendo, el proceso de normalización hasta 4FN

1. Encontrar las dependencias funcionales
2. Encontrar las claves candidatas
3. Verificar si el esquema cumple con la definición de BCNF, si no se cumple descomponer la relación sin perder información ni dependencias funcionales
 - Si se pierden dependencias funcionales, llevar a 3NF
 - Esto hasta dejar las particiones en BCNF o 3FN
4. Expresar en este punto, cual es la clave primaria y que particiones quedaron en BCNF o 3FN (según corresponda)
5. Encontrar las dependencias multivaluadas sobre la última partición realizada (aquella que tiene la clave primaria del esquema) y verificar 4NF, si no se cumple dividir la relación
 - Esto se hace hasta dejar las particiones del esquema en 4FN
6. Expresar las particiones resultantes que quedaron en 4FN
 - Explicar porque las particiones descritas en el ítem 4 (excepto la analizada en el punto 5) quedaron en 4FN
7. Indicar que particiones en 4FN quedan en el esquema final (que no sean proyecciones de atributos claves de otras particiones en 4FN)

Lunes 09/09
Publicación TP2





Bibliografía de la clase

Bibliografía

- Date, C. J. (2019). *Database design and relational theory: normal forms and all that jazz*. Apress.
- Garcia-Molina, H. (2008). *Database systems: the complete book*. Pearson Education India.
- Ullman, J. D. (1988). Principles of database and knowledge-base systems.
- Albarak, M., Bahsoon, R., Ozkaya, I., & Nord, R. L. (2020). Managing Technical Debt in Database Normalization. *IEEE Transactions on Software Engineering*.
- Jadhav, R., Dhabe, P., Gandewar, S., Mirani, P., & Chugwani, R. (2020). A New Data Structure for Representation of Relational Databases for Application in the Normalization Process. In *Machine Learning and Information Processing* (pp. 305-316). Springer, Singapore.
- Ghawi, R. (2019, May). Interactive Decomposition of Relational Database Schemes Using Recommendations. In *International Conference: Beyond Databases, Architectures and Structures* (pp. 97-108). Springer, Cham.
- Stefanidis, C., & Koloniari, G. (2016, November). An interactive tool for teaching and learning database normalization. In *Proceedings of the 20th Pan-Hellenic Conference on Informatics* (pp. 1-4).
- Knowledge Base of Relational and NoSQL Database Management Systems https://db-engines.com/en/ranking_trend
- Akhtar, A. (2023). Popularity Ranking of Database Management Systems. *arXiv preprint arXiv:2301.00847*.



Importante!
Los slides usados en las clases teóricas de esta materia, no son material de estudio por sí solos en ningun caso.