Apunte de Normalización para los alumnos de la cátedra de Bases de Datos 1 Cursada 2013

En el diseño de bases de datos existen propiedades indeseables que no deberían presentarse en los esquemas. Éstas son repetición de información, mala representación de la información y pérdida de la misma.

- **Repetición de Información**

  Supongamos que se tiene el siguiente esquema (llamado R) que representa los alumnos y las materias a las que se inscriben (un alumno se puede inscribir en varias materias):

  R(dni,nomYApe,dir,edad,codMateria,nombreMateria)

  Considerando este esquema para cada nueva materia en la que se inscriba un alumno se tendrá repetido el nombre, el apellido, la dirección y la edad. Así como para cada alumno que se inscribe a la misma materia, el nombre de la misma se repetirá. Veamos con un ejemplo esta situación.

  Tenemos a los alumnos Pedro y Juan que se inscriben a las materias bases de datos 1 y bases de datos 2, en el esquema R esto se reflejaría con la inserción de las siguientes tuplas:

Dni	nomYApe	dir	edad	codMateria	nombreMateria
30468459	Pedro	7 y 50	25	C1	BBDD1
30468459	Pedro	7 y 50	25	C2	BBDD2
32065456	Juan	8 y 55	23	C1	BBDD1
32065456	Juan	8 y 55	23	C2	BBDD2

Tabla 1: Inserción en el esquema R

  Como se ve en la tabla 1, cada vez que inserto una nueva inscripción para el alumno Pedro repito el nombre y apellido, la dirección y su edad. Lo mismo pasa con la información insertada para el alumno Juan. Notar además que con cada inscripción realizada, se repite el nombre de la materia a la que se está inscribiendo.
  Supongamos ahora que el alumno Pedro cambia su dirección. Esto implicaría buscar cada una de las tuplas que correspondan a este alumno y modificar este dato. Si esto no se hace se produce inconsistencia de datos puesto que el mismo alumno en diferentes tuplas figura con diferentes domicilios.

- **Problema en la Representación de la Información**

  Veamos los problemas que puede traer una mala representación de un esquema. Supongamos que se cuenta con el esquema R definido anteriormente:

R(dni,nomYApe,dir,edad,codMateria,nombreMateria)

Dicho esquema no permite insertar un alumno sin asociarle al menos una inscripción a una materia. Ahora bien, cuando un alumno inicia una carrera, no es cierto que éste se inscriba a una materia en particular, por ejemplo porque las inscripciones a materias se abren más adelante en el tiempo.
Ante este escenario planteado, que soluciones intuitivas se podrían plantear?:

- ¿Cargo valores nulos en atributos?
- ¿Creo un código de material especial al que interpreto como que el alumno en realidad no esta inscripto a ninguna materia aun?

En estos casos que parecieran ser soluciones, en realidad estoy introduciendo en mi modelo de datos una solución provisoria para corregir una mala decisión de diseño del esquema R.

- **Pérdida de la Información**

Veamos un ejemplo de cómo se puede perder información. Supongamos que se cuenta con el esquema R definido previamente

R(dni,nomYApe,dir,edad,codMateria,nombreMateria)

Una división intuitiva para el esquema R seria la siguiente:

R1(dni,nomYApe,dir,edad)
R2(codMateria,nombreMateria)
R3(dni,codMateria)

Pero que pasa si intuitivamente se hace una mala división. Supongamos que se divide el esquema **R** de la siguiente manera:

R4(dni, nomYApe,dir,edad)
R5(codMateria,nombreMateria)
R6(nomYApe,codMateria)

En este caso la intuición no sirvió y se logro perder la información de las materias en las que se inscribió un alumno, ya que la edad puede ser compartida por varios alumnos.

Para evitar las características no deseables en el diseño de los esquemas, surge la idea de normalización que permite hacer divisiones de un esquema evitando estos problemas. La división del esquema se puede realizar empleando el concepto de dependencia funcional. Cuando normalizo un esquema no uso la intuición sino que aplico un proceso ya establecido.

Conceptos relacionados con Normalización:

- Dependencias funcionales
- Super clave
- Clave candidata
- Clave primaria
- Atributo primo
- Formas normales

- **Dependencias funcionales**

Las dependencias funcionales sirven para dar semántica a las tablas y para definir restricciones sobre las mismas. Una dependencia funcional significa que para un X dado siempre se recupera el mismo valor de Y. Una dependencia funcional se denota de la siguiente manera:

$$X \rightarrow Y$$

Una dependencia funcional de la forma X → Y se cumple en un esquema **R** si para todos los pares de tuplas t1 y t2 de una relación r, tal que t1[x]=t2[x], entonces t1[y]=t2[y], donde t[x] es el valor del atributo x para la tupla t.

Una dependencia funcional es considerada trivial si se da la siguiente condición.

X → Y es trivial si Y es un subconjunto de X ($Y \subseteq X$).

X → Y es una *dependencia funcional total* si la eliminación de cualquier atributo A de X hace que la dependencia funcional deje de ser válida.

X → Y es una *dependencia funcional parcial* si la eliminación de cualquier atributo A de X hace que la dependencia funcional siga siendo válida.

Ejemplo:

Supongamos que se tiene el siguiente esquema:

ALUMNO(dni,nomYApe,fechaNacimiento)

Dado un *dni* se puede determinar un único nombre y apellido con una única fecha de nacimiento de un alumno. Una posible dependencia funcional del esquema puede ser:

dni → nomYApe, fechaNacimiento

A partir del esquema se puede especificar la siguiente dependencia funcional trivial:

dni, nomYApe → dni

En este caso dni ⊆ (dni, nomYApe) por esta condición es considerada una dependencia funcional trivial. Un ejemplo de dependencia funcional parcial del esquema puede ser:

dni, nomYApe → fechaNacimiento

En este caso si se saca el atributo *nomYApe* la dependencia funcional sigue valiendo por eso es considerada una dependencia funcional parcial. Un ejemplo de dependencia funcional total del esquema puede ser:

dni → fechaNacimiento

En este caso si se saca el atributo *dni* la dependencia funcional deja de valer, por tal motivo es una dependencia funcional total.

Para el proceso de normalización, buscaremos que las dependencias funcionales halladas sean totales.

Clausura de un conjunto de dependencias funcionales (F^+)

A partir de un conjunto de dependencias funcionales de un esquema **R** es posible deducir nuevas dependencias funcionales válidas en ese mismo esquema. Este conjunto completo se halla mediante los axiomas de Amstrong (estos axiomas son enunciados mas adelante en el documento)
Supongamos que se tiene la dependencia funcional a → b se sabe por la definición de dependencia funcional que hay un par de tuplas t1 y t2 tal que t1[a]=t2[a], entonces t1[b]=t2[b]. Si además se tiene la dependencia b→c, aplicando la definición de dependencia funcional se tiene que t1[c]= t2[c]. Por transitividad se obtiene a→c. Es decir,:

a → b implica t1[a]=t2[a] → t1[b]=t2[b]

b → c implica t1[b]=t2[b] → t1[c]=t2[c]

Por transitividad se tiene:

a → c implica t1[a]=t2[a] → t1[c]=t2[c]

Ejemplo:

Supongamos que se tienen el siguiente esquema:

R(a,b,c,g,h,i)

Con las siguientes dependencias funcionales sobre **R:**

a → b
a → c
c,g → h
c,g → i
b → h

Supongamos que se busca la clausura de este conjunto de dependencias funcionales (F^+). Se sabe por definición de dependencia funcional lo siguiente:

a → b tal que t1[a]=t2[a] → t1[b]=t2[b]

b → h tal que t1[b]=t2[b] → t1[h]=t2[h]

Por transitividad se obtiene una nueva dependencia funcional:

a → h tal que t1[a]=t2[a] → t1[h]=t2[h]

Por lo tanto, la clausura del conjunto de dependencias funcionales del esquema queda definida con las siguientes dependencias funcionales:

a → b
a → c
c,g → h
c,g → i
b → h
a → h

Axiomas de Amstrong

Permiten inferir nuevas dependencias funcionales dado un conjunto base de dependencias funcionales que resulto evidente. Aplicando los axiomas se encuentra un conjunto completo y seguro donde todas las dependencias funcionales halladas son correctas. Luego de aplicar los axiomas pueden quedar dependencias funcionales triviales.

- **Axiomas Básicos:**
  - **Reflexibidad**

    X es un conjunto de atributos y Y⊆ X entonces X → Y

    Es decir, si Y⊆ X y existen dos tuplas diferentes en el esquema R, llamémoslas t1 y t2, tales que t1[x]=t2[x] por definición de dependencia funcional t1[y]=t2[y]

    Ejemplo:

Si se tiene (a,b) y a $\subseteq$ (a,b) luego se puede decir que a,b → a

- **Aumento**

  Si X → Y y Z es un conjunto de atributos, entonces Z,X → Z,Y

  Ejemplo:

  Si se tiene a → d, y el atributo c entonces vale a,c → d,c

- **Transitividad**

  Si se tiene X →Y y Y → Z, entonces se tiene la dependencia funcional X → Z

  Se puede decir que si se tiene X → Y entonces t1[x]=t2[x] implica t1[y]=t2[y] y si además se tiene Y → Z entonces t1[y]=t2[y] implica t1[z]=t2[z]. Por transitividad se tiene X → Z entonces t1[x]=t2[x] implica t1[z]=t2[z].

  Ejemplo:

  Si se tiene a → b y b → c entonces vale a → c

- **Axiomas que se deducen a partir de los axiomas básicos:**

  - **Unión**

    Si se tienen X →Y y X → Z, entonces se tiene X → Y,Z

    Si se tiene la dependencia funcional X → Y y la dependencia funcional X → Z. Por aumento vale que X → XY. Además vale por aumentación X,Y → Y,Z. Luego por transitividad vale X → Y,Z.

    Ejemplo:

    Si se tiene a → b y a → c entonces vale a → b,c

  - **Descomposición**

    Si se tiene X → Y,Z, entonces X → Y y X → Z

    Si se tiene X →Y,Z. Por reflexividad vale que Y,Z →Y. Luego, por transitividad vale X→Y. Por otro lado, por reflexividad también vale que Y,Z → Z. Luego por transitividad, también vale que X→Z.

    Ejemplo:

Si se tiene a → b,c entonces vale a → b y a → c

- **Pseudotransitividad**

  Si se tiene X →Y y Y,Z→W entonces X,Z → W

  Si se tiene X →Y por aumento vale que X,Z →Y,Z. Por otro lado se sabe que Y,Z→W, por transitividad vale que X,Z->W.

  Ejemplo:

  Si se tiene a → b y b,c → d entonces vale a,c → d

Análisis de Pérdida de Información

Para corroborar que no se pierda información al particionar un esquema **R** en dos subesquemas $\mathbf{R_1}$ y $\mathbf{R_2}$ se debe cumplir alguna de las siguientes condiciones:

$\mathbf{R_1} \cap \mathbf{R_2}$ es clave en el esquema $\mathbf{R_1}$

O bien

$\mathbf{R_1} \cap \mathbf{R_2}$ es clave en el esquema $\mathbf{R_2}$

Y además se debe cumplir que $\mathbf{R_1} \cup \mathbf{R_2} = \mathbf{R}$ {esto asegura que no se perdieron atributos}

Análisis de pérdida de dependencias funcionales cuando se particiona un esquema R.

Cuando se particiona un esquema **R**, además de no perder información, se debe tener en cuenta que no se pierdan dependencias funcionales. Para ello se debe verificar que cada una de las dependencias funcionales que valían en el esquema **R**, sigan valiendo en alguna de las particiones.

Cuando se chequean las dependencias funcionales pueden ocurrir dos cosas:

a) los atributos de la dependencia funcional original quedaron todos incluidos en alguna de las particiones generadas
b) los atributos de la dependencia funcional original quedaron distribuidos en mas de una partición

Ejemplos de particionamiento y análisis de pérdida de dependencias funcionales:

Ejemplo 1)

Sea **R1** un esquema y **F** el conjunto de dependencias funcionales que valen en **R1**

R1(a,b,c,d)
F = {a→b, b→c, c→d}

Particionamos el esquema **R1** de la siguiente manera:

R1.1(a,b)
R1.2(b,c)
R1.3(c,d)

En este caso, como las dependencias funcionales valen en **R1** y la división es una división sin pérdida de información, se puede decir que:

a→b vale en **R1.1**
b→c vale en **R1.2**
c→d vale en **R1.3**

ya que los atributos de las dependencias funcionales originales (las descriptas en el conjunto F) quedaron todos incluidos en alguna de las particiones generadas

Ejemplo 2)

Sea **R2** un esquema y **F** el conjunto de dependencias funcionales que valen en **R2**

R2(a,b,c,d)

Con las siguientes dependencias funcionales:

F = {a → b, b → c, c → d, d → a}

Particionamos el esquema **R2** de la siguiente manera:

R2.1(a,b)
R2.2(b,c)
R2.3(c,d)

En este caso, se puede decir que:

a→b vale en **R2.1**
b→c vale en **R2.2**
c→d vale en **R2.3**

pero no se puede decir nada de la dependencia funcional d→a

Cuando una dependencia funcional queda *cruzada* como en el ejemplo anterior, se debe verificar si la misma se sigue cumpliendo en la partición generada. Es decir, el hecho de que una dependencia funcional quede cruzada no necesariamente significa que no se cumpla más, y que en consecuencia se perdió.

Desde el punto de vista formal, verificar una dependencia funcional que quedó cruzada (como en el ejemplo anterior) significa:

- Calcular F^+ (es decir la clausura de un conjunto de dependencias funcionales) sobre el esquema **R**.
- Calcular $G = \bigcup_{i=1,k} DF_i$ tal que $atributosDF_i \subseteq R_i$

  Donde:

  - R un esquema
  - F el conjunto de dependencias funcionales válidas en el esquema R
  - DF_i es una dependencia funcional (de la forma X -> Y) incluída en F^+
  - $atributosDF_i$ el conjunto de atributos de la dependencia DF_i, es decir (X,Y)
  - k el número de particiones realizado sobre un esquema R
  - R_i es una de las k particiones de R

Es decir G va a ser el conjunto de todas las dependencias funcionales que valen sobre **R** y cuyos atributos quedaron todos en una misma partición.

Una vez calculado G, se calcula G^+, si en G^+ aparece la dependencia funcional d→a entonces la dependencia se preserva.

A continuación se siguen estos pasos para verificar si la dependencia funcional d→a sigue valiendo en la partición del ejemplo 2.

Como se mencionó anteriormente se realizan las siguientes particiones del esquema **R2**:

R2.1(a,b)
R2.2(b,c)
R2.3(c,d)

con el siguiente conjunto de dependencias funcionales:

F= {a→b, b→c, c→d, d→a}

Ahora se calcula F^+, entre otro conjunto de dependencias que se encuentran se puede ver:

(Notar que en F^+ solamente se contemplan las dependencias funcionales que son relevantes al ejemplo)

F^+={ a→b, b→c, c→d, b→a, c→b, d→a, d→c, …}

Estas dependencias funcionales se calculan a partir las siguiente transitividades:

b→c y c→d y d→a, luego b→a
c→d y d→a y a→b, luego c→b
d→a y a->b y b→c, luego d→c

Ahora se debe armar el conjunto G con los datos que se tienen:

G= {a→b, b →c, c→d, b→a, c→b, d→c}

Notar que en el conjunto G no se incluye la dependencia funcional d→a ya que no aparece directamente en ninguna R_i.

Una vez obtenido el conjunto G, se halla la clausura de G, para ver si puedo deducir la dependencia funcional d→a (dependencia funcional cuyos atributos están en dos particiones).
Por transitividad, llego a que:

d→c y c→b y b→a, luego d→a.

Luego, podemos decir que la dependencia funcional d→a se preserva en la partición propuesta.

Si bien en el ejemplo propuesto, se pudieron simplificar los conjuntos F^+ y G $^+$, calcularlos no siempre es trivial.
Para evitar calcular las clausuras de los conjuntos de dependencias funcionales F y G en cada caso, existe un algoritmo que permite verificar la preservación de la dependencia funcional. Dicho algoritmo se presenta a continuación.

Algoritmo para analizar la pérdida de dependencias funcionales

Res = x
Mientras Res cambia
Para i= 1 to cant_de_ particiones_realizadas
Res = Res ∪ ((Res ∩ Ri)$^+$ ∩ Ri)

Donde:

- R_i es el conjunto de atributos de la división representada por R_i

- X es el determinante de la dependencia funcional que quiero analizar.
- $((Res \cap Ri)^+ \cap Ri)$, asegura que quedan sólo los atributos que pertenecen a la partición que se esta tratando.

Para calcular $(Res \cap Ri)^+$ se debe aplicar el algoritmo que se da a continuación para hallar la clausura de atributos (Notar que el funcionamiento de este algoritmo se especificará más adelante, ahora simplemente se enuncia para poder desarrollar el ejemplo).

```
result:= X
While (hay cambios en result) do
   For (cada dependencia funcional de la forma ( Y →Z) en F ) do
                  if (Y ⊆ result) then
                     result := result ∪ Z
```

Ejemplo:

Supongamos que se tiene el siguiente esquema:

R2(a,b,c,d)

Con el siguiente conjunto de dependencias funcionales:

F= {a->b, b ->c, c->d, d->a}

Supongamos que se realizan las siguientes particiones del esquema **R2**:

R2.1(a,b)
R2.2(b,c)
R2.3(c,d)

Con el algoritmo para analizar pérdida de dependencias funcionales, se puede verificar si se perdió la dependencia funcional d→a.

Res = x
Mientras Res cambia
Para i= 1 to cant_de_ particiones_realizadas
Res = Res ∪ ((Res ∩ Ri)$^+$ ∩ Ri)

A continuación se ejecuta el algoritmo partiendo de Res= d :

Paso i=1)

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

Paso i=2)

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

Paso i=3)

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

Se termina la iteración y Res cambio, entonces se comienza nuevamente.

Paso i=1)

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

Paso i=2)

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

Paso i=3)

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

Se termina la iteración y Res cambio, entonces se comienza nuevamente.

Paso i=1)

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

Res = {c,d,b,a}

En este paso se ve que partiendo del atributo d, se recupera el atributo a, entonces la dependencia funcional d → a ***NO SE PERDIO***.

- **Superclave**

Una superclave es un conjunto K de atributos, los cuales conjuntamente, identifican unívocamente a una entidad en el conjunto de entidades. Si K es superclave cualquier conjunto que contenga a K es superclave.

Ejemplo:

Supongamos que se tiene el siguiente esquema:

ALUMNO(dni,nomYApe,fechaNacimiento)

Se puede decir que *dni* es superclave. También se puede decir que *dni, nomYApe* es superclave ya que es un super conjunto que contiene a la superclave.

- **Clave candidata**

Si R es un esquema de relación con atributos $A_1,A_2,\ldots A_n$ y dependencias funcionales F y además X es un subconjunto de atributos de R, decimos que X es clave si:

1) La dependencia funcional $X \rightarrow A_1,\ldots,A_n$ esta en F^+
2) No hay ningún subconjunto propio $Y \subset X$, tal que $Y \rightarrow A_1,\ldots,A_n$ (esto asegura la minimalidad).

Una clave candidata es una superclave en la cual ningún subconjunto propio es superclave. Puede existir más de una clave candidata.

Ejemplo:

Supongamos que se tiene el siguiente esquema:

ALUMNO(dni,nomYApe,fechaNacimiento,#Alumno)

Se puede decir que tanto *dni* como *#Alumno* son claves candidatas.

- **Clave primaria**

Cuando un esquema R tiene mas de una clave candidata, en el proceso de normalización queda sólo una de ellas y las otras quedan divididas en el esquema, sin que esto implique pérdida de información.
Esa clave candidata que queda luego del proceso, se denomina clave primaria.

Ejemplo:

Supongamos que se tiene el siguiente esquema:

ALUMNO(dni,nomYApe,fechaNacimiento,#Alumno)

Se puede decir que tanto *dni* como *#Alumno* son claves candidatas. Supongamos que luego del proceso de normalización, queda como clave primaria *dni*.

- **Atributo primo**

Un atributo del esquema de relación R se denomina atributo primo de R si es miembro de cualquier clave candidata de R.

Ejemplo:

Supongamos que se tiene el siguiente esquema:

R(#nroalumno, dni, #materia)

Donde se sabe que un alumno cursa muchas materias. Las claves candidatas (cc) de este esquema son:

cc1:(dni, #materia)
cc2: (#nroalumno, #materia)

En este caso #materia es un atributo primo porque es miembro de las dos claves candidatas del esquema.

- **Clausura de un conjunto de atributos (X^+)**

Sea F un conjunto de dependencias funcionales sobre un conjunto de atributos llamado U y sea X un subconjunto de U. La clausura de X respecto de F, se denota X^+ y es el conjunto de atributos A tal que la dependencia funcional X → A puede deducirse a partir de F por los axiomas de Amstrong.

Algoritmo para encontrar X^+

Con este algoritmo se puede verificar si un conjunto de atributos X de un esquema R conforman una posible clave candidata del esquema R.

```
Result:= X
While (hay cambios en result) do
        For (cada dependencia funcional de la forma (Y → Z) en F) do
                if (Y ⊆ result) then
```

result := result ∪ Z

Ejemplos de aplicación:

Ejemplo 1)

Supongamos que se tiene el siguiente esquema **R** para ejecutar el algoritmo para encontrar X^+.

R(a,b,c,g,h,i)

Con el siguiente conjunto F de dependencias funcionales:

F={
a→b,c
c,g →h, i
b→h
}

Supongamos que se propone como clave candidata (a,g), entonces hay que buscar la clausura de atributos (a,g) para verificar que se pueden encontrar todos los demás atributos a partir de estos. Veamos los pasos de ejecución del algoritmo para hallar la clausura de atributos:

Result= a,g
Paso 1)
Se toma la dependencia funcional a→b,c
a esta incluido en result
agrego b y c a result => (a,g,b,c)

Paso 2)
Se toma la dependencia funcional c,g →h, i
(c,g) esta incluido en result
agrego h e i a result => (a,g,b,c,h, i)

Paso 3)
Se toma la dependencia funcional b →h
b esta incluido en result, h también.

Se corta el while porque se hallaron todos los atributos de **R** (comprobando que (a,g) es una posible clave candidata).

Ejemplo 2)

Veamos otro ejemplo de ejecución del algoritmo para hallar la clausura de atributos.
Supongamos que se tiene el mismo esquema que en el ejemplo anterior pero ahora se agrega una dependencia funcional más (h → a).

R(a,b,c,g,h,i)

Con el siguiente conjunto F de dependencias funcionales:

F={
a→b,c
c,g →h,i
b→h
h → a
}

Supongamos que se propone como clave candidata (c,g), entonces hay que buscar la clausura de atributos (c,g) para verificar que se pueden encontrar todos los demás atributos a partir de estos.
Veamos los pasos de ejecución del algoritmo.

Result= c,g
Paso 1)
Se toma la dependencia funcional a → b,c
a no esta incluido en result
result no cambia

Paso 2)
Se toma la dependencia funcional c,g → h, i
(c,g) esta incluido en result
agrego h e i a result => (c,g,h,i)

Paso 3)
Se toma la dependencia funcional b → h
b no esta incluido en result
result no cambia

Paso 4)
Se toma la dependencia funcional h → a
h esta incluído en result
agrego a a result => (c,g,h,i,a)

Como result ha cambiado y no se pudieron determinar todos los atributos de R, se hace otra iteración.

Paso 5)

Se toma la dependencia funcional a → b, c
a esta incluído en result
agrego b y c a result => (c,g,h,i,a,b, c)

Paso 6)

Se toma la dependencia funcional c,g →h,i
h e i ya estan incluidos
result no cambia

Paso 7)

Se toma la dependencia funcional b→h
h ya esta incluido
result no cambia

Paso 8)

Se toma la dependencia funcional h → a
a ya esta incluido
result no cambia

Como ya se determinaron a todos los atributos de **R** se termina la iteración (comprobando que (c,g) es una posible clave candidata).

Formas Normales (FN)

1FN

Se prohíben los atributos multivaluados (aquellos que tienen múltiples valores), los atributos compuestos (aquellos compuestos a su vez de otros atributos) y sus combinaciones.

2FN:

Se basa en el concepto de dependencia funcional total. Una dependencia funcional X → Y es una dependencia funcional total si la eliminación de cualquier atributo A de X hace que la dependencia funcional deje de ser válida. Una relación R está en 2FN si está en 1FN y si todo atributo no primo depende funcionalmente de manera total de la clave primaria de R.

Ejemplo:

Supongamos que se tiene el siguiente esquema:

> **EMP_PROYECTO**(**dni**,nombreApellido,horasTrabajadas,**#Proyecto**,
> nombreProyecto, lugarProyecto)

La clave candidata de este esquema es: **dni, #Proyecto**

Hallo las dependencias funcionales del esquema:

1. dni, #Proyecto → horasTrabajadas
2. dni → nombreApellido
3. #Proyecto → nombreProyecto, lugarProyecto

Al ver las dependencias funcionales, veo que el *nombreApellido* se recupera sin necesidad de considerar el #proyecto en el antecedente de la dependencia funcional 2, es decir que el atributo no primo *nombreApellido* no depende de manera total de la clave candidata. Entonces existe una dependencia funcional parcial. Puedo decir que el esquema **EMP_PROYECTO** no esta en 2NF.

3FN:

Se basa en el concepto de dependencia funcional transitiva. Una dependencia funcional X →Y es una dependencia funcional transitiva si existe un conjunto de atributos Z que no sea un subconjunto de cualquier clave de R y se cumplen tanto X → Z como Z → Y.
Un esquema esta en 3NF si esta en 2NF y ningún atributo no primo de R depende transitivamente de la clave primaria. Es decir, un esquema esta en 3NF siempre que una dependencia funcional X → A se cumple en R, o bien:

a) X es superclave de R, o bien,
b) A es un atributo primo de R

Ejemplo:

Dado el siguiente esquema:

EMP_DEPTO(nombreApellido, **dni**, fechaNac, direccion, #Depto, nombreDepto)

Se sabe que un empleado trabaja únicamente en un departamento, y que en cada departamento trabajan varios empleados.

La clave candidata es *dni*. Se tienen las siguientes dependencias funcionales:

1. dni → nomApellido, fechaNac, direccion, #Depto

2. #Depto → nombreDepto

La tabla **EMP_DEPTO** no esta en 3NF ya que existe la dependencia funcional 2 que no cumple con ninguna de las dos condiciones de 3NF. Es decir:

- *#Depto* no es superclave en R
- *nombreDepto* no es un atributo primo de **EMP_DEPTO**

Forma normal de Boyce-Cod (BCNF)

Es más restricta que la 3FN, lo que significa que toda relación que este en BCNF estará en 3NF pero no necesariamente una relación en 3NF esta en BCNF.
Un esquema de relación esta en BCNF si, siempre que una dependencia funcional X→A es válida en R, entonces X es superclave de R. Es decir, la única diferencia entre BCNF y 3FN es que en BCNF no esta presente la condición b) de 3NF.

Ejemplo:

Supongamos que se tiene el siguiente esquema:

R(#nroalumno, dni, #materia)

Donde se sabe que un alumno cursa muchas materias. Las claves candidatas (cc) de este esquema son:

cc1:(dni, #materia)
cc2: (#nroalumno, #materia)

Se tienen las siguientes dependencias funcionales:

1. #nroalumno → dni
2. dni → #nroalumno

R no esta en BCNF ya que al menos *#nroalumno* no es superclave de **R**.

4FN:

Se basa en el concepto de dependencia multivaluada.Un esquema esta en 4FN cuando:
- no existen dependencias multivaluadas o
- sólo existe una dependencia multivaluada trivial

Nota: Los conceptos para definir y analizar 4 FN serán publicados luego que los mismos sean abordados en las clases teóricas.

Proceso usado para la normalización de las tablas

Este proceso se debe realizar cuando se necesita normalizar hasta 4FN un esquema dado, considerando que ya se encuentra en 1FN.

El proceso de normalización para un esquema R, usado y evaluado en la materia consta de los siguientes pasos:

1- Identificar en el esquema R la o las claves candidatas y escribirlas explícitamente poniendo los atributos que conforman cada una de ellas.
2- Identificar las dependencias funcionales no triviales y **mínimas** válidas en el esquema R y escribirlas explícitamente numerándolas en orden consecutivo. Tener en cuenta que el orden en la que se escriben no será necesariamente el orden en el que posteriormente se usen en el paso 3 (LOS PASOS 1 Y 2 NO MANTIENEN UN ORDEN ESTRICTO, se pueden invertir)
3- Iniciar el análisis del esquema R para normalizarlo hasta BCNF (o 3FN). Comprobando si el esquema R se halla en BCNF.
    a. Si no esta en BCNF, entonces intentar llevarlo a BCNF creando particiones del esquema R (ver **Proceso de particionamiento de un esquema analizando desde BCNF**)
    b. Si esta en BCNF detener el proceso
4- Una vez finalizado el proceso escribir las particiones resultantes que se generan en el punto 3 (esquemas que están en BCNF o 3FN según corresponda)
5- Indicar la clave primaria resultante del proceso de normalización (tener en cuenta que debe haber quedado como clave primaria, alguna de las claves candidatas halladas en el punto 1)
6- Una vez identificadas las tablas que luego de aplicar el proceso de particionamiento de un esquema analizando desde BCNF quedaron en BCNF (o 3FN) analizar 4FN
    i. Si el proceso se realizo de acuerdo a lo propuesto en la materia, la única tabla que ***puede*** no estar en 4FN es la última que se hallo con el proceso.
        a. En caso de detectar que un esquema no se encuentra en 4FN, se deberá aplicar *un proceso que será incluido en otro documento una vez introducido este concepto en las clases teóricas.*
    ii. Una vez llevadas las particiones a 4FN, escribir de manera explicita al final de todo el proceso cuales son las tablas que considera que han quedado en 4FN y no son proyecciones de otras tablas.

Supongamos ahora que una vez aplicados los pasos 1 y 2 del proceso, al realizar el paso 3, se detecta un esquema que no se encuentra en BCNF. Dada esta condición, se debe realizar el siguiente proceso de particionamiento:

Proceso de particionamiento de un esquema analizando desde BCNF

Cuando un esquema R no cumple la condición de BCNF, es decir existe en R al menos una dependencia funcional de la forma X→A tal que X no es superclave de R. Se debe proceder a la partición del esquema R. Para esto:

a) se genera una partición R_i conteniendo los atributos de la dependencia funcional X→A
b) se genera una partición R_{i+1} con los atributos de R – A. Esta forma de particionar asegura que no se pierda información ya que:
   - $\mathbf{R_i \cap R_{i+1}}$ seguro es X por construcción, esto permite la relación entre los dos esquemas, y además
   - $\mathbf{R_i \cup R_{i+1} = R}$

¡Importante! En el caso de tener sobre R dependencias funcionales mínimas donde algunas de ellas son transitivas (al menos uno de los atributos del antecedente de una dependencia funcional es al menos uno de los atributos del consecuente de otra dependencia funcional), por ejemplo:

a,d-> b,e
b,f->c

se deberá tratar primero la dependencia funcional mínima cuyo antecedente contiene al menos uno de los atributos del consecuente de la otra dependencia funcional, en el ejemplo planteado, primero se debe tratar la dependencia funcional b,f->c y luego la dependencia funcional a,d-> b,e.
Si no se trata en el orden establecido previamente, la dependencia funcional se pierde por el orden en el que se tratan las mismas, sin que esto implique que el esquema no se pueda llevar a BCNF.

Una vez generadas las particiones se debe:

- Analizar la partición Ri indicando que dependencias funcionales mínimas de las indicadas sobre el esquema R valen en Ri. Indicar cual es la clave primaria del esquema Ri considerando la dependencia funcional mínima por la que se particionó el esquema original. Si las dependencias funcionales que quedaron valiendo en la partición Ri son mínimas, Ri quedo en BCNF.
  - **¡Importante!**
    - Si las dependencias funcionales planteadas sobre R no son mínimas, podría pasar que en Ri aparecieran nuevas dependencias funcionales que hagan que el esquema Ri no quedara en BCNF siendo necesario analizar BCNF en el esquema Ri. Para realizar este análisis es necesario

reiniciar el proceso de normalización desde el punto 3 ahora para el esquema R_i

- Si las dependencias funcionales halladas son mínimas pero no halle todas las claves candidatas del esquema y en consecuencia todas las dependencias funcionales válidas teniendo en cuenta esto, pueden surgir nuevas dependencias funcionales mínimas en R_i y seguir valiendo la definición de BCNF en R_i

- Analizar la partición R_{i+1} de modo que en ella queden las dependencias funcionales restantes aun no tratadas del esquema R. Es decir, aquellas dependencias funcionales que no quedaron siendo válidas en la partición R_i
  - Si están todas las dependencias funcionales, entonces marcar en la partición R_{i+1} las claves candidatas válidas en R y volver a analizar si ahora el esquema R_{i+1} se encuentra en BCNF. Para realizar este análisis es necesario reiniciar el proceso de normalización desde el punto 3 ahora para el esquema R_{i+1}
  - Si se detecta que hay atributos de una dependencia funcional que quedaron distribuidos en las particiones R_i y R_{i+1} analizar si se perdió la dependencia funcional usando el ***algoritmo de pérdida de dependencias funcionales*** para validarlo.
    - Si se perdió la dependencia funcional al particionar el esquema R para llevar a BCNF entonces, el esquema R no puede ser llevado a BCNF y se debe analizar 3 FN (en el caso de que BCNF falla y se debe analizar 3FN en el esquema R ver **Proceso de particionamiento de un esquema cuando falla BCNF y se debe analizar 3FN**). Luego de este análisis se pasa al punto 4 del proceso.
    - Si no se perdió ninguna dependencia funcional, se procede a analizar la partición R_{i+1}. En este punto, se vuelve a analizar si ahora el esquema R_{i+1} se encuentra en BCNF. Para realizar este análisis es necesario reiniciar el proceso de normalización desde el punto 3 ahora para el esquema R_{i+1}

      Aclaración: Si se detecta que la partición R_{i+1} contiene sólo el conjunto de atributos que conforman alguna de las claves candidatas halladas para el esquema R, se puede afirmar que el esquema R_{i+1} ya esta en BCNF terminando así el proceso.

Una vez finalizado el paso 3 del proceso, se pasa al punto 4 continuando con el proceso hasta el punto 6.

Proceso de particionamiento de un esquema cuando falla BCNF y se debe analizar 3FN

A este punto de análisis se llega cuando desde el proceso de particionamiento de un esquema a BCNF, sucede que se perdió una dependencia funcional [1]al particionar el esquema R en las particiones R_i y R_{i+1} ,entonces, el esquema R no puede ser llevado a BCNF y se debe analizar 3 FN sobre el esquema R.

Cuando esto sucede, existen dependencias funcionales que valen en el esquema original a normalizar y que aún no han sido tratadas y dejadas en tablas intermedias normalizadas a BCNF o 3 FN.

Para llevar el esquema R a 3FN

- *Si existe una única clave candidata válida en el esquema original a normalizar:*

  Se debe crear para cada una de estas dependencias funcionales válidas en R una R_i. En caso de que la clave candidata no haya quedado en ninguna de las particiones R_i se debe agregar una partición solo con el conjunto de atributos de la clave candidata.

- *Si existe mas de una clave candidata en el esquema original a normalizar:*

  Al existir más de una clave candidata, hay dependencias funcionales que se corresponden con una clave candidata y otras que se corresponden con las restantes claves candidatas. Se toma una clave candidata como clave primaria, se contemplan las dependencias funcionales que se corresponden con esta decisión y se particiona como en el caso anterior.

[1] *Nota: Tener en cuenta que antes de poder asegurar que se perdió una dependencia funcional, se debe mirar si cambiando el orden en el que se tratan las dependencias funcionales, la "supuesta" pérdida de la dependencia funcional no se subsana.*

Ejemplos de normalización usando el proceso de normalización para un esquema R, usado y evaluado en la materia

Ejemplo 1: Se aplica el *proceso de normalización sobre el esquema FIESTA, y en el proceso a BCNF no falla (es decir no hay que analizar 3FN)*

Supongamos que tenemos el siguiente esquema:

FIESTA (#Salon, dirección, capacidad, fechaFiesta, nomContratante, cantInvitados, nombreInvitado, cantMesas, mesaInvitado, servicioContratado, dirContratante, dniInvitado)

- En cada salón se realiza una sola fiesta por día
- En un día puede haber varias fiestas en diferentes salones.
- Para cada fiesta puede figurar más de un contratante.
- En distintas fiestas el mismo contratante puede figurar con diferentes direcciones.
- Cada invitado tiene asociado un número de mesa.
- La cantidad de mesas del salón varía para cada fiesta.
- Servicio contratado es una lista (En la tabla aparecerá una tupla por cada servicio contratado) que describe los tipos de comida contratados para una fiesta.
- Una persona puede ir a más de una fiesta en el mismo salón.

Paso 1) Marcar en el esquema FIESTA la o las claves candidatas(CC)

CC: (#Salon, fechaFiesta, dniInvitado, nombreContratante, servicioContratado)

Paso 2) Escribir las dependencias funcionales no triviales y mínimas válidas en el esquema FIESTA

1. #Salon → dirección, capacidad
2. dniInvitado→ nombreInvitado
3. #Salon, fechaFiesta, nombreContratante → dirContratante
4. #Salon, fechaFiesta, dniInvitado → mesaInvitado
5. #Salon, fechaFiesta → cantInvitados, cantMesas

Paso 3) Iniciar el análisis del esquema FIESTA para normalizarlo hasta BCNF (o 3FN). Comprobando si el esquema FIESTA se halla en BCNF.

El esquema **FIESTA** no esta en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 1-5 no es superclave en el esquema **FIESTA** (Por ejemplo: la dependencia funcional 1). Entonces divido la tabla utilizando la dependencia funcional 1 quedando lo siguiente:

F1(#Salon, dirección, capacidad)
F2(#Salon, fechaFiesta, nombreContratante, cantInvitados, nombreInvitado, cantMesas, mesaInvitado, servicioContratado, dirContratante, dniInvitado)

El esquema **F1** está en BCNF ya que el antecedente de la dependencia funcional 1 es superclave en **F1** y vale la dependencia funcional 1 la cual se uso en el proceso de división. El esquema **F2** no esta en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 2-5 no es superclave en **F2** (Por ejemplo: la dependencia funcional 2). Entonces divido la tabla **F2** utilizando la dependencia funcional 2 quedando lo siguiente:

F3(dniInvitado, nombreInvitado)
F4(#Salon, fechaFiesta, nombreContratante, cantInvitados, cantMesas, mesaInvitado, servicioContratado, dirContratante, dniInvitado)

El esquema **F3** esta en BCNF ya que el antecedente de la dependencia funcional 2 es superclave de **F3** y vale la dependencia funcional 2 la cual se uso en el proceso de división. El esquema **F4** no esta en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 3-5 no es superclave de **F4** (Por ejemplo: la dependencia funcional 3). Entonces divido la tabla **F4** utilizando la dependencia funcional 3 quedando lo siguiente:

F5(#Salon, fechaFiesta, nombreContratante, dirContratante)
F6(#Salon, fechaFiesta, nombreContratante, cantInvitados, cantMesas, mesaInvitado, servicioContratado, dniInvitado)

El esquema **F5** esta en BCNF ya que el antecedente de la dependencia funcional 3 es superclave de **F5** y vale la dependencia funcional 3 la cual se uso en el proceso de división. El esquema **F6** no esta en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 4 y 5 no es superclave de **F6** (Por ejemplo: la dependencia funcional 4). Entonces divido la tabla **F6** utilizando la dependencia funcional 4 quedando lo siguiente:

F7(#Salon, fechaFiesta, dniInvitado, mesaInvitado)
F8(#Salon,fechaFiesta,nombreContratante,cantInvitados,cantMesas,servicioContratado, dniInvitado)

El esquema **F7** esta en BCNF ya que el antecedente de la dependencia funcional 4 es superclave de **F7** y vale la dependencia funcional 4 la cual se uso en el proceso de división. El esquema **F8** no esta en BCNF porque el antecedente de la dependencia funcional 5 no es superclave de **F8**. Entonces divido la tabla **F8** utilizando la dependencia funcional 5 quedando lo siguiente:

F9(#Salon, fechaFiesta,cantInvitados, cantMesas)
F10(#Salon fechaFiesta,nombreContratante, servicioContratado, dniInvitado)

El esquema **F9** esta en BCNF ya que el antecedente de la dependencia funcional 5 es superclave de **F9** y vale la dependencia funcional 5 la cual se uso en el proceso de división. El esquema **F10** esta en BCNF porque todos los atributos forman parte de la clave de dicho esquema.

Paso 4) Una vez finalizado el proceso escribir las particiones resultantes que se generan en el punto 3 (esquemas que están en BCNF o 3FN según corresponda)

Tablas en BCNF: F1, F3, F5, F7, F9 y F10.

Paso 5) Indicar la clave primaria resultante del proceso de normalización (tener en cuenta que debe haber quedado como clave primaria, alguna de las claves candidatas halladas en el punto 1)

Clave Primaria:(#Salon,fechaFiesta,dniInvitado,nombreContratante, servicioContratado)

Paso 6) Una vez identificadas las tablas que luego de aplicar el proceso de particionamiento de un esquema analizando desde BCNF quedaron en BCNF (o 3FN) analizar 4FN

(el análisis a 4 FN será publicado luego de que los conceptos de 4 FN sean impartidos en la teoría)

Ejemplo 2: Se aplica el proceso de normalización sobre el esquema R, y por las características halladas, se debe aplicar el Proceso de particionamiento de un esquema cuando falla BCNF y se debe analizar 3FN

Supongamos que tenemos el siguiente esquema:

R(a,b,c,d,e,f,g)

Paso 1) Marcar en el esquema R la o las claves candidatas(CC)

CC: (d,e,a)

Paso 2) Escribir las dependencias funcionales no triviales y mínimas válidas en el esquema R

1. a → b,c
2. d,e → f
3. e → g
4. a → g

Paso 3) Iniciar el análisis del esquema R para normalizarlo hasta BCNF (o 3FN). Comprobando si el esquema R se halla en BCNF.

El esquema **R** no esta en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 1-4 no es superclave de **R** (Por ejemplo: la dependencia funcional 1). Entonces divido la tabla utilizando la dependencia funcional 1 quedando lo siguiente:

R1(**a**,b,c)
R2(**d,e**,f,g,**a**)

El esquema **R1** está en BCNF ya que el antecedente de la dependencia funcional 1 es superclave de **R1** y vale la dependencia funcional 1 la cual se uso en el proceso de división. El esquema **R2** no esta en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 2-4 no es superclave de **R2** (Por ejemplo: la dependencia funcional 2). Entonces divido la tabla **R2** utilizando la dependencia funcional 2 quedando lo siguiente:

R3(**d,e**,f)
R4(**d,e,a**,g)

El esquema **R3** está en BCNF ya que el antecedente de la dependencia funcional 2 es superclave en **R3** y vale la dependencia funcional 2 la cual se uso en el proceso de división. El esquema **R4** no esta en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 3 y 4 no es superclave de **R4** (Por ejemplo: la dependencia funcional 4). Entonces divido la tabla **R4** utilizando la dependencia funcional 4 quedando lo siguiente:

R5(**a**,g)
R6(**a,d,e**)

- Se debe analizar si se perdió la dependencia funcional e → g

Para esto aplico el algoritmo para determinar pérdida de dependencias funcionales

RES: (e)
Mientras Res cambia
Para i= 1 to cant_de_ particiones_realizadas
Res = Res ∪ ((Res ∩ Ri)$^+$ ∩ Ri)

Desde i = 1 hasta 4 (Son las particiones R1-R3-R5-R6)

Paso 1)

R1(a,b,c)

Res = (e) ∪ (((e) ∩ (a,b,c))$^+$ ∩ (a,b,c))
Res = (e)

Paso 2)

R3(d,e,f)
Res = (e) ∪ (((e) ∩ (d,e,f))$^+$ ∩ (d,e,f))
Res = (e) ∪ ((e)$^+$ ∩ (d,e,f))

-Debo hallar la clausura del conjunto de atributos (e)
Result:= X
While (hay cambios en result) do
For (cada dependencia funcional de la forma (Y → Z) en F) do
if (Y ⊆ result) then
result := result ∪ Z

result : (e)
para cada dependencia funcional del conjunto
{ a → b,c
d,e → f
e → g
a → g}
Entra al while por primera vez y result= (e,g)

Como result cambio, entro una vez mas al while
result : (e,g)
para cada dependencia funcional del conjunto
{ a → b,c
d,e → f
e → g
a → g }
En esta iteración, result queda igual y termino la iteración
result : (e,g)

Res = (e) ∪ ((e,g) ∩ (d,e,f))
Res = (e) ∪ (e)
Res = (e)

Paso 3)

R5(a,g)
Res = (e) ∪ (((e) ∩ (a,g))$^+$ ∩ (a,g))
Res = (e)

Paso 4)

R6(a,d,e)
Res = (e) ∪ (((e) ∩ (a,d,e))$^+$ ∩ (a,d,e))
Res = (e) ∪ ((e)$^+$ ∩ (a,d,e))

-Debo hallar la clausura del conjunto de atributos (e)
Result:= X
While (hay cambios en result) do
For (cada dependencia funcional de la forma (Y → Z) en F) do
if (Y ⊆ result) then
result := result ∪ Z

result : (e)
para cada dependencia funcional del conjunto
{ a → b,c
d,e → f
e → g
a → g}
Entra al while por primera vez y result= (e,g)

Como result cambio, entro una vez mas al while
result : (e,g)
para cada dependencia funcional del conjunto
{ a → b,c
d,e → f
e → g
a → g }
En esta iteración, result queda igual y termino la iteración
result : (e,g)

Res = (e) ∪ ((e,g) ∩ (a,d,e))
Res = (e) ∪ (e)
Res = (e)

Con la aplicación del algoritmo de pérdida de dependencias funcionales, se demuestra que con esta división se pierde la dependencia funcional e → g, entonces hay aplicar el proceso de particionamiento de un esquema cuando falla BCNF analizando 3FN.
Dado que sólo existe una clave candidata, se arma una tabla con cada una de las dependencias funcionales que quedan y otra con la clave.

R4.1(**e**,g)
R4.2(**a**,g)
R4.3(**d,e,a**)

Paso 4) Una vez finalizado el proceso escribir las particiones resultantes que se generan en el punto 3 (esquemas que están en BCNF o 3FN según corresponda) **Tablas en BCNF:** R1, R3

Tablas en 3FN: R4.1, R4.2 y R4.3

Paso 5) Indicar la clave primaria resultante del proceso de normalización (tener en cuenta que debe haber quedado como clave primaria, alguna de las claves candidatas halladas en el punto 1)

Clave Primaria: (d,e,a)

Paso 6) Una vez identificadas las tablas que luego de aplicar el proceso de particionamiento de un esquema analizando desde BCNF quedaron en BCNF (o 3FN) analizar 4FN

Conceptos para analizar la 4 forma normal

Dependencia multivaluada

Sea R un esquema de relación. La dependencia multivaluada X -->> Y vale en **R** si ∀ los pares de tuplas t1 y t2 en R, tal que t1[X] = t2[X] existen las tuplas t3 y t4 en R tales que:

t1[X] = t2[X] = t3[X] = t4[X]
t3[Y] = t1[Y]
t3[R-X-Y] = t2[R-X-Y]
t4[Y] = t2[Y]
t4[R-X-Y] = t1[R-X-Y]

En otras palabras se puede decir que: **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**.
Una dependencia multivaluada de la forma **X->> Y**, es trivial cuando el conjunto de atributos {**X**,**Y**} conforma el total de los atributos del esquema.

Ejemplos:

En los ejemplos que analizaremos, usaremos un conjunto de datos para mostrar los valores del esquema y así introducir la noción práctica de dependencias multivaluadas. Este análisis que se propone desde la visualización de los datos, en realidad surge de la interpretación de las restricciones propuestas para el esquema a analizar.

Ejemplo 1) Supongamos que se tiene el siguiente esquema que está en BCNF:

INFORMACION_PAIS(#Pais, #Prov, atractivoTuristico)

Sobre el esquema **INFORMACION_PAIS** valen las siguientes restricciones:

- Un *#Pais* tiene varias provincias.
- Una provincia puede estar en diferentes países.
- Los atractivos turísticos son todos los atractivos de un país. El mismo atractivo puede estar en más de un país.

Un conjunto de datos posibles para el esquema **INFORMACION_PAIS** es el siguiente:

#Pais	#Prov	atractivoTuristico
Argentina	Misiones	Cataratas del Iguazú
Argentina	Buenos Aires	Cataratas del Iguazú
Argentina	Mendoza	Cataratas del Iguazú
Argentina	Misiones	Puerto Madero
Argentina	Buenos Aires	Puerto Madero
Argentina	Mendoza	Puerto Madero
Argentina	Misiones	Aconcagua
Argentina	Buenos Aires	Aconcagua
Argentina	Mendoza	Aconcagua

Tabla 2

Se puede observar en la **Tabla 1** que para un valor de país, por ejemplo, tengo muchos valores para provincia. Y para un país se tienen muchos atractivos turísticos. Esto mismo que se ve con los datos se deduce del conjunto de restricciones dado.

Supongamos que queremos ver si la siguiente dependencia multivaluada propuesta es válida:

#Pais -->> #Prov

Se debe ver si existe independencia del atributo #Prov respecto de los demás atributos del esquema que no están involucrado en la dependencia multivaluada propuesta para verificar si ésta es válida. Para probar esto se toma un *#Pais* fijo y se verifica que todos los valores de *#Prov* (para dicho *#Pais*) estén combinados con los demás atributos no incluidos en la dependencia multivaluada propuesta, en este caso *atractivoTuristico (es decir existe un producto cartesiano entre los valores de #Prov y atractivoTuristico para un valor de #País fijo)*.

Si se toma *#Pais = Argentina* todos los valores de *#Prov* son: Misiones, Buenos Aires y Mendoza. Estos valores de provincias están combinados con todos los valores de *atractivoTuristico* (Cataratas del Iguazú, Puerto Madero, Aconcagua). Por lo tanto hay independencia de atributos y la dependencia multivaluada

#Pais -->> #Prov es válida.

Supongamos que ahora se quiere probar si es válida la dependencia multivaluada dada por:

#Pais -->> atractivoTurístico.

De acuerdo al razonamiento expuesto anteriormente, si se toma *#Pais = Argentina* todos los valores de *atractivoTurístico* son: *Cataratas del Iguazú, Puerto Madero y Aconcagua*. Estos valores de provincias están combinados con todos los valores de *#Prov* (*Misiones, Buenos Aires, Mendoza*). Por lo tanto hay independencia de atributos y la dependencia multivaluada

#Pais -->> atractivoTurístico es válida.

Ejemplo 2) Supongamos que se tiene el siguiente esquema que está en BCNF:

INFO_PAIS(#Pais, #Prov, atractivoTuristico)

Sobre el esquema **INFO_PAIS** valen las siguientes restricciones:

- Un *#Pais* tiene varias provincias.
- Una provincia puede estar en diferentes países.
- *Los atractivos turísticos son todos los atractivos de una provincia dentro de un país.* El mismo atractivo turístico puede estar en más de una provincia.

Notar que si bien el esquema es el mismo que se propuso en el ejemplo 1), al cambiar las restricciones sobre el esquema, la forma de los datos cambia y las dependencias multivaluadas, también.

Un conjunto de datos posibles para el esquema **INFO_PAIS** es el siguiente:

#Pais	#Prov	atractivoTuristico
Argentina	Prov1	Cataratas del Iguazú
Argentina	Prov2	Puerto Madero
Argentina	Prov3	Aconcagua
Argentina	Prov2	Recoleta
Argentina	Prov2	La Boca
Brasil	Prov1	Cataratas del Iguazú

Tabla 2

Se puede observar en la **Tabla 2** que para una provincia dentro de un país se tienen muchos atractivos turísticos. Esto mismo que se ve con los datos se deduce del conjunto de restricciones dado.

Supongamos que queremos ver si es válida la dependencia multivaluada dada por

#Pais -->> #Prov

Se debe ver si existe independencia respecto de los demás atributos para verificar si la dependencia multivaluada es válida. Para probar esto se toma un *#Pais* fijo y se verifica que todos los valores de *#Prov* (para dicho *#Pais*) este combinados con los demás atributos no incluidos en la dependencia multivaluada, en este caso *atractivoTuristico*.

Se puede apreciar en la **Tabla 2** que para *#Pais = Argentina* todos los valores de *#Prov* (*Prov1, Prov2, Prov3*) no están combinados con todos los valores de *atractivoTuristico* (*Cataratas del Iguazú, Puerto Madero, Aconcagua, Recoleta, La Boca*). Supongamos *#Prov = Prov3* no está combinado con *Recoleta* ni con *La Boca.* Esto quiere decir que no hay independencia de atributos por lo tanto la dependencia multivaluada

#Pais -->> #Prov no es válida

para el esquema **INFO_PAIS.** Esto sucedió porque está mal planteada la dependencia multivaluada respecto de las restricciones **INFO_PAIS.** No existe independencia del atributo #Prov en relación al atributo atractivoTuristico.

Considerando las restricciones planteadas para el esquema **INFO_PAIS** se plantea la siguiente dependencia multivaluada:

#Pais, #Prov -->> atractivoTurístico

Está dependencia respeta las restricciones planteadas para el esquema **INFO_PAIS**. Y además es una dependencia trivial en dicho esquema. Está dependencia multivaluada surge de la siguiente restricción planteada para el esquema:

- Los atractivos turísticos son todos los atractivos de una provincia dentro de un país. El mismo atractivo turístico puede estar en más de una provincia.

Luego, podemos decir que la dependencia multivaluada:

#Pais, #Prov -->> atractivoTurístico es válida

Ejemplo 3) Supongamos que se tiene el siguiente esquema que está en BCNF:

INFORMACION_CURSO (#Curso,material,profesor)

Sobre el esquema **INFORMACION_CURSO** valen las siguientes restricciones:

- Los cursos tienen varios profesores asignados.
- Un curso tiene asignado muchos materiales. Los materiales son propios del curso. Cuando un profesor dicta un curso ya los tiene disponibles

Un conjunto de datos posibles para el esquema **INFORMACION_CURSO** es el siguiente:

#Curso	profesor	material
17	Eva	1
17	Eva	2
17	Julia	1
17	Julia	2
25	Eva	1
25	Eva	2
25	Eva	3

Tabla 3

Se puede observar en la **Tabla 3** que para un valor de curso se tienen muchos profesores asignados. Y para un curso se tienen muchos materiales que son usados por todos los profesores que dictan ese curso.
Supongamos que queremos ver si es válida la dependencia multivaluada dada por:

#Curso -->> material

Se debe ver si existe independencia respecto de los demás atributos para verificar si la dependencia multivaluada es válida. Para probar esto se toma un *#Curso* fijo y se verifica que todos los valores de *material* (para dicho *#Curso*) este combinados con los demás atributos no incluidos en la dependencia multivaluada, en este caso *profesor*.

Según los valores dados en la **Tabla 3**. Las tuplas que se analizan tomando *#Curso = 17* son las siguientes:

#Curso	profesor	material
17	Eva	1
17	Eva	2
17	Julia	1
17	Julia	2

Si se toma *#Curso = 17* todos los valores de *material* son: 1 y 2. Estos valores de *material* están combinados con todos los valores de *profesor* (Eva y Julia).

Si ahora se toma *#Curso = 25,* las tuplas que se analizan son las siguientes:

#Curso	profesor	material
25	Eva	1
25	Eva	2
25	Eva	3

Si se toma *#Curso = 25* todos los valores de *material* son: 1, 2 y 3. Estos valores de *material* están combinados con todos los valores de *profesor* (Eva). Por lo tanto hay independencia de atributos y la dependencia multivaluada

#Curso -->> material es válida.

Supongamos que ahora se quiere probar si es válida la dependencia multivaluada dada por:

#Curso -->> profesor

Según los valores dados en la **Tabla 3**. Las tuplas que se analizan tomando *#Curso = 17* son las siguientes:

#Curso	profesor	material
17	Eva	1
17	Eva	2
17	Julia	1
17	Julia	2

Si se toma *#Curso = 17* todos los valores de *profesor* son: Eva y Julia. Estos valores de *profesor* están combinados con todos los valores de *material* (*1y 2).*

Si ahora se toma *#Curso = 25,* las tuplas que se analizan son las siguientes:

#Curso	profesor	material
25	Eva	1
25	Eva	2

25	Eva	3

Si se toma *#Curso = 25* todos los valores de *profesor* es solamente: Eva. Este valor de *profesor* está combinado con todos los valores de *material* (*1y 2).* Por lo tanto hay independencia de atributos y la dependencia multivaluada

#Curso -->> profesor es válida.

Supongamos que ahora se quiere probar si es válida la dependencia multivaluada dada por:

profesor-->> *#Curso*

Según los valores dados en la **Tabla 3**. Las tuplas que se analizan tomando *profesor = Eva* son las siguientes:

#Curso	profesor	material
17	Eva	1
17	Eva	2
25	Eva	1
25	Eva	2
25	Eva	3

Si se toma *profesor = Eva* todos los valores de *#Curso* son: 17 y 25. Estos valores de *#Curso* **no** están combinados con todos los valores de *material* (*1, 2, 3).* En el conjunto de datos no está la tupla *(17, Eva, 3)*. La dependencia multivaluada profesor-->> #Curso está mal planteada respecto de las restricciones e interpretaciones establecidas para el esquema **INFORMACION_CURSO.** No existe independencia del atributo #Curso en relación al atributo material.

Ejemplo 4) Supongamos que se tiene el siguiente esquema que está en BCNF:

CURSO (#Curso,material,profesor)

Sobre el esquema **CURSO** valen las siguientes restricciones:

- Los cursos tienen varios profesores asignados.
- *Cada profesor dentro de un curso usa distintos materiales didácticos.*

Un conjunto de datos posibles para el esquema **CURSO** es el siguiente:

#Curso	profesor	material
17	Eva	1
17	Eva	2
17	Julia	3

17	Julia	1
25	Eva	1
25	Eva	2
25	Julia	3

Tabla 4

Se puede observar en la **Tabla 4** que los cursos tienen varios profesores asignados. Y que cada profesor dentro de un curso usa distintos materiales didácticos.

Supongamos que queremos ver si es válida la dependencia multivaluada dada por:

#Curso -->> material

Se debe ver si existe independencia respecto de los demás atributos para verificar si la dependencia multivaluada es válida. Para probar esto se toma un *#Curso* fijo y se verifica que todos los valores de *material* (para dicho *#Curso*) este combinados con los demás atributos no incluidos en la dependencia multivaluada, en este caso *profesor*.

Según los valores dados en la **Tabla 4**. Las tuplas que se analizan tomando *#Curso = 17* son las siguientes:

#Curso	profesor	material
17	Eva	1
17	Eva	2
17	Julia	3
17	Julia	1

Se puede que para *#Curso = 17* todos los valores de *material* (*1, 2, 3*) no están combinados con todos los valores de *profesor (Eva y Julia).*
Supongamos *material = 2* no está combinado con *Julia.* Esto quiere decir que no hay independencia de atributos por lo tanto la dependencia multivaluada

#Curso -->> material no es válida

para el esquema **CURSO.** Esto sucedió porque está mal planteada la dependencia multivaluada respecto de las restricciones para **CURSO.**

Planteamos considerando las restricciones la siguiente dependencia multivaluada para el esquema **CURSO**:

#Curso, profesor -->> material

Está dependencia respeta las restricciones planteadas para el esquema **CURSO**. Y además es una dependencia trivial en dicho esquema. No hay atributos restantes a la dependencia multivaluada para verificar independencia de atributos. Luego, la dependencia multivaluada:

#Curso, profesor -->> material es válida

Ejemplo 5) Supongamos que se tiene el siguiente esquema que está en BCNF:

PROFESOR_CURSO(#Curso,material,profesor)

Sobre el esquema **PROFESOR_CURSO** valen las siguientes restricciones:

- Los cursos tienen varios profesores asignados.
- Un profesor usa materiales independientemente del curso. Esto quiere decir que el material es usado por cada profesor en cada curso que dicta

Un conjunto de datos posibles para el esquema **PROFESOR_CURSO** es el siguiente:

#Curso	profesor	material
17	Eva	1
17	Eva	2
17	Julia	1
17	Julia	3
25	Eva	1
25	Eva	2

Tabla 5

Se puede observar en la **Tabla 5** que los cursos tienen varios profesores asignados. Un profesor usa diferentes materiales independientemente del curso.

Supongamos que queremos ver si es válida la dependencia multivaluada dada por:

#Curso -->> profesor

Se debe ver si existe independencia respecto de los demás atributos para verificar si la dependencia multivaluada es válida. Para probar esto se toma un *#Curso* fijo y se verifica que todos los valores de *profesor* (para dicho *#Curso*) este combinados con los demás atributos no incluidos en la dependencia multivaluada, en este caso *material*.

Según los valores dados en la **Tabla 5**. Las tuplas que se analizan tomando *#Curso = 17* son las siguientes:

#Curso	profesor	material
17	Eva	1
17	Eva	2
17	Julia	1
17	Julia	3

Se puede que para #Curso = *17* todos los valores de *profesor* (*Eva y Julia*) no están combinados con todos los valores de *material (1, 2 y 3).* Supongamos *profesor = Eva* no está combinado con el material 3. Esto quiere decir que no hay independencia de atributos por lo tanto la dependencia multivaluada

#Curso -->> profesor no es válida

para el esquema **PROFESOR_CURSO.** Esto sucedió porque está mal planteada la dependencia multivaluada respecto de las restricciones para **CURSO.**

Supongamos que queremos ver si es válida la dependencia multivaluada dada por:

profesor -->> material

Según los valores dados en la **Tabla 5**. Las tuplas que se analizan tomando *profesor = Eva* son las siguientes:

#Curso	profesor	material
17	Eva	1
17	Eva	2
25	Eva	1
25	Eva	2

Se puede apreciar que para *profesor = Eva* todos los valores de *material* (*1, 2*) están combinados con todos los valores de *#Curso (17 y 25).*
Si ahora se toma *profesor = Julia,* las tuplas que se analizan son las siguientes:

#Curso	profesor	material
17	Julia	1
17	Julia	3

Se puede observar que si se toma *profesor = Julia* todos los valores de *material* (*1, 3*) están combinados con todos los valores de *#Curso (17).*

Por lo tanto hay independencia de atributos y la dependencia multivaluada

profesor -->> material es válida

Caso especial de Dependencia Multivaluada (vacio multidetermina un atributo):

Ejemplo 6) Supongamos que se tiene el siguiente esquema que está en BCNF:

VIDEO_CLUB (#Cliente, #Video)

Sobre el esquema **VIDEO_CLUB** valen las siguientes restricciones:

- #Cliente representa a todos los clientes del video club.
- #Video representa a cada uno de los videos existentes en el video club.

Un conjunto de datos posibles para el esquema **VIDEO_CLUB** es el siguiente:

#Cliente	#Video
C1	V1
C1	V2
C1	V3
C2	V1
C2	V2
C2	V3

Tabla 6

Se tiene en la **Tabla 6** información de los clientes y los videos del video club pero estos atributos no tienen ninguna relación entre si según las restricciones establecidas para el esquema **VIDEO_CLUB.**

Analicemos si son válidas las dependencias multivaluadas sobre el esquema **VIDEO_CLUB**:

∅ -->> #Cliente
∅ -->> #Video

Supongamos que queremos ver si es válida la dependencia multivaluada dada por:

∅ -->> #Cliente

Se debe ver si existe independencia respecto de los demás atributos para verificar si la dependencia multivaluada es válida. Para probar las dependencias multivaluadas determinadas por el vacio se verifica que todos los valores de #Cliente este combinados con los demás atributos no incluidos en la dependencia multivaluada, en este caso #Video.

Se puede apreciar en la **Tabla 6**, todos los valores de *#Cliente* (*C1 y C2*) están combinados con todos los valores de *#Video (V1, V2 y V3).* Esto quiere decir que hay independencia de atributos por lo tanto la dependencia multivaluada

∅ -->> #Cliente es válida

para el esquema **VIDEO_CLUB.**

Supongamos que queremos ver si es válida la dependencia multivaluada dada por:

∅ -->> #Video

Se puede apreciar en la **Tabla 6**, que todos los valores de *#Video* (*V1, V2 y V3)* están combinados con todos los valores de *#Cliente (C1 y C2*)*).* Esto quiere decir que hay independencia de atributos por lo tanto la dependencia multivaluada

∅ -->> #Video es válida

para el esquema **VIDEO_CLUB.**

Ejemplo 7) Supongamos que se tiene el siguiente esquema que está en BCNF:

CURSO_MATERIAL_PROFESOR (#Curso,material,profesor)

Sobre el esquema **CURSO_MATERIAL_PROFESOR** valen las siguientes restricciones:

- *#Curso* representa todos los cursos de la unidad académica.
- *material* representa todos los materiales usados en la unidad académica.
- *profesor* representa todos los profesores de la unidad académica.

Un conjunto de datos posibles para el esquema **CURSO_MATERIAL_PROFESOR** es el siguiente:

#Curso	profesor	material
17	Eva	1
17	Eva	2
17	Eva	3
17	Julia	1
17	Julia	2
17	Julia	3
25	Eva	1
25	Eva	2
25	Eva	3
25	Julia	1
25	Julia	2
25	Julia	3

Tabla 7

Analicemos si valen las siguientes dependencias multivaluadas sobre esquema **CURSO_MATERIAL_PROFESOR**:

∅ -->> #Curso
∅ -->> material
∅ -->> profesor

Supongamos que queremos ver si es válida la dependencia multivaluada dada por:

∅ -->> #Curso

Se debe ver si existe independencia respecto de los demás atributos para verificar si la dependencia multivaluada es válida. Para probar las dependencias multivaluadas determinadas por el vacio se verifica que todos los valores de *#Curso* estén combinados con los demás atributos no incluidos en la dependencia multivaluada, en este caso *material* y *profesor*.

Se puede apreciar en la **Tabla 7**, todos los valores de *#Curso* (*17 y 25*) están combinados con todos los valores de *profesor* y *material.* Es decir, los cursos 17 y 25 combinados con las tuplas (Eva, 1), (Eva, 2), (Eva, 3), (Julia, 1), (Julia, 2) y (Julia, 3). Esto quiere decir que hay independencia de atributos por lo tanto la dependencia multivaluada

∅ -->> #Curso es válida

para el esquema **CURSO_MATERIAL_PROFESOR.**

Supongamos que queremos ver si es válida la dependencia multivaluada dada por:

∅ -->> material

Todos los valores de profesor (*Eva* y *Julia*) están combinados con todos los valores de *#Curso y profesor.* Es decir, los materiales 1, 2 y 3 están combinados con las tuplas (17, Eva), (17, Julia), (25, Eva), y (25, Julia). Esto quiere decir que hay independencia de atributos por lo tanto la dependencia multivaluada

∅ -->> material es válida

para el esquema **CURSO_MATERIAL_PROFESOR.**

Supongamos que queremos ver si es válida la dependencia multivaluada dada por:

∅ -->> profesor

Todos los valores de material (*1, 2 y 3*) están combinados con todos los valores de *#Curso y material.* Es decir, los profesores *Eva* y *Julia* están combinados con las tuplas (17, 1), (17, 2), (17, 3), (25, 1), (25, 2) y (25, 3). Esto quiere decir que hay independencia de atributos por lo tanto la dependencia multivaluada

∅ -->> profesor es válida

para el esquema **CURSO_MATERIAL_PROFESOR.**

Ejemplo 8) Supongamos que se tiene el siguiente esquema que está en BCNF:

UNIDAD_ACADEMICA (#Curso,material, profesor, #Alumno)

Sobre el esquema **UNIDAD_ACADEMICA** valen las siguientes restricciones:

- Los cursos tienen varios profesores asignados.
- Cada profesor dentro de un curso usa distintos materiales didácticos.
- *#Alumno* representa a cada uno de los alumnos de la unidad académica.

Un conjunto de datos posibles para el esquema **CURSO_MATERIAL_PROFESOR** es el siguiente:

#Curso	profesor	material	#Alumno
17	Eva	1	A1
17	Eva	2	A1
17	Julia	1	A1
17	Julia	3	A1
25	Eva	1	A1
25	Eva	2	A1
17	Eva	1	A2
17	Eva	2	A2
17	Julia	1	A2
17	Julia	3	A2
25	Eva	1	A2
25	Eva	2	A2

Tabla 8

Analicemos si valen las siguientes dependencias multivaluadas sobre esquema **UNIDAD_ACADEMICA**:

1. #Curso, profesor -->> material
2. ∅ -->> #Alumno

Supongamos que queremos ver si es válida la dependencia multivaluada dada por:

∅ -->> #Alumno

Se debe ver si existe independencia respecto de los demás atributos para verificar si la dependencia multivaluada es válida. Para probar las dependencias multivaluadas determinadas por el vacio se verifica que todos los valores de *#Alumno* estén combinados con los demás atributos no incluidos en la dependencia multivaluada, en este caso *#Curso*, *material* y *profesor*.

Se puede apreciar en la **Tabla 8**, todos los valores de #Alumno (*A1 y A2*) están combinados con todos los valores de *#Curso*, *material* y *profesor.* Es decir, los #Alumno A1 y A2 están

combinados con las tuplas (17, Eva, 1), (17, Eva, 2), (25, Eva, 1), (25, Eva, 2) (17, Julia, 1) y (17, Julia, 3). Esto quiere decir que hay independencia de atributos por lo tanto la dependencia multivaluada

∅ -->> #Alumno es válida

para el esquema **UNIDAD_ACADEMICA.**

Supongamos que queremos ver si es válida la dependencia multivaluada dada por:

#Curso, profesor -->> material

Según los valores dados en la **Tabla 8**. Las tuplas que se analizan tomando *#Curso, profesor = 17, Eva* son las siguientes:

#Curso	profesor	material	#Alumno
17	Eva	1	A1
17	Eva	2	A1
17	Eva	1	A2
17	Eva	2	A2

Se puede apreciar que para *#Curso, profesor = 17, Eva* todos los valores de *material* (*1, 2*) están combinados con todos los valores de *#Alumno (A1 y A2).*

Supongamos que ahora se toma fijo *#Curso, profesor = 17, Julia* quedan las siguientes tuplas para analizar son las siguientes:

#Curso	profesor	material	#Alumno
17	Julia	1	A1
17	Julia	3	A1
17	Julia	1	A2
17	Julia	3	A2

Se puede apreciar que para *#Curso, profesor = 17, Julia* todos los valores de *material* (*1, 3*) están combinados con todos los valores de *#Alumno (A1 y A2).*

Supongamos que ahora se toma fijo *#Curso, profesor = 25, Eva* quedan las siguientes tuplas para analizar son las siguientes:

#Curso	profesor	material	#Alumno
25	Eva	1	A1
25	Eva	2	A1
25	Eva	1	A2
25	Eva	2	A2

Se puede apreciar que para *#Curso, profesor = 25, Eva* todos los valores de *material* (*1, 2*) están combinados con todos los valores de *#Alumno (A1 y A2).*

Esto quiere decir que hay independencia de atributos por lo tanto la dependencia multivaluada

#Curso, profesor -->> material es válida

para el esquema **UNIDAD_ACADEMICA.**

4ta. Forma Normal

Se basa en el concepto de dependencia multivaluada.
Un esquema **R** está en 4FN cuando para cada dependencia multivaluada en R de la forma X→> Y, se cumple alguna de las condiciones siguientes:

1) X→> Y es trivial
2) X es superclave de R

Una **dependencia multivaluada es trivial** cuando el conjunto de atributos X,Y son todos los atributos del esquema.

*Nota: Tener en cuenta que si un esquema **R** no posee dependencias multivaluadas, ese esquema se encuentra en 4FN.*

Ejemplos:

Ejemplo 1) Supongamos que se tiene el siguiente esquema que está en BCNF:

INFORMACION_PAIS(#Pais, #Prov, atractivoTuristico)

Sobre el esquema **INFORMACION_PAIS** valen las siguientes restricciones:

- Un *#Pais* tiene varias provincias.
- Una provincia puede estar en diferentes países.
- Los atractivos turísticos son todos los atractivos de un país. El atractivo turístico se puede repetir en diferentes países.

En el esquema **INFORMACION_PAIS** valen las siguientes dependencias multivaluadas:

1. #Pais -->> #Prov
2. #Pais -->> atractivoTurístico

El esquema **INFORMACION_PAIS** no está en 4FN porque existen las dependencias multivaluadas 1 y 2 que no son triviales en dicho esquema.

Ejemplo 2) Supongamos que se tiene el siguiente esquema que está en BCNF:

INFORMACION_CURSO (#Curso,material,profesor)

Sobre el esquema **INFORMACION_CURSO** valen las siguientes restricciones:

- Los cursos tienen varios profesores asignados.
- Un curso tiene asignado muchos materiales. Los materiales son propios del curso. Cuando un profesor dicta un curso ya los tiene disponibles

En el esquema **INFORMACION_CURSO** valen las siguientes dependencias multivaluadas

1. #Curso -->> material
2. #Curso -->> profesor

El esquema **INFORMACION_CURSO** no está en 4FN porque existen las dependencias multivaluadas 1 y 2 que no son triviales en dicho esquema.

Ejemplo 3) Supongamos que se tiene el siguiente esquema que está en BCNF:

CURSO (#Curso,material,profesor)

Sobre el esquema **CURSO** valen las siguientes restricciones:

- Los cursos tienen varios profesores asignados.
- cada profesor dentro de un curso usa distintos materiales didácticos.

En el esquema **CURSO** vale la siguiente dependencia multivaluada:

1. #Curso, profesor -->> material

El esquema **CURSO** está en 4FN porque solo vale la dependencia multivaluada 1 que es trivial en dicho esquema.

Ejemplo 4) Supongamos que se tiene el siguiente esquema que está en BCNF:

VIDEO_CLUB (#Cliente, #Video)

Sobre el esquema **VIDEO_CLUB** valen las siguientes restricciones:

- #Cliente representa a todos los clientes del video club.
- #Video representa a cada uno de los videos existentes en el video club.

En el esquema **VIDEO_CLUB** valen las siguientes dependencias multivaluadas:

1. ∅ -->> #Cliente

2. ∅ -->> #Video

El esquema **VIDEO_CLUB** no está en 4FN porque existen las dependencias multivaluadas 1 y 2 que no son triviales en dicho esquema.

Ejemplo 5) Supongamos que se tiene el siguiente esquema que está en BCNF:

UNIDAD_ACADEMICA (#Curso,material,profesor, #Alumno)

Sobre el esquema **UNIDAD_ACADEMICA** valen las siguientes restricciones:

- Los cursos tienen varios profesores asignados.
- Cada profesor dentro de un curso usa distintos materiales didácticos.
- #Alumno representa a cada uno de los alumnos de la unidad académica.

En el esquema **UNIDAD_ACADEMICA** valen las siguientes dependencias multivaluadas:

1. #Curso, profesor -->> material
2. ∅ -->> #Alumno

El esquema **UNIDAD_ACADEMICA** no está en 4FN porque existen las dependencias multivaluadas 1 y 2 que no son triviales en dicho esquema.

Ejemplo 6) Supongamos que se tiene el siguiente esquema que está en BCNF:

PROFESOR_CURSO(#Curso,material,profesor)

Sobre el esquema **PROFESOR_CURSO** valen las siguientes restricciones:

- Los cursos tienen varios profesores asignados.
- Un profesor usa materiales independientemente del curso. Esto quiere decir que el material es usado por cada profesor en cada curso que dicta

En el esquema **PROFESOR_CURSO** vale la siguiente dependencia multivaluada:

1. profesor -->> material

El esquema **PROFESOR_CURSO** no está en 4FN porque existe la dependencia multivaluada 1 que no es trivial en dicho esquema.

Una vez introducidos los conceptos para analizar 4FN, se presenta el proceso de normalización de manera completa hasta 4 FN.

Proceso completo usado para la normalización de las tablas

Este proceso se debe realizar cuando se necesita normalizar hasta 4FN un esquema dado, considerando que ya se encuentra en 1FN.

3. Identificar en el esquema R la o las claves candidatas y escribirlas explícitamente poniendo los atributos que conforman cada una de ellas.
4. Identificar las dependencias funcionales no triviales y **mínimas** válidas en el esquema R y escribirlas explícitamente numerándolas en orden consecutivo. Tener en cuenta que el orden en la que se escriben no será necesariamente el orden en el que posteriormente se usen en el paso 3 (LOS PASOS 1 Y 2 NO MANTIENEN UN ORDEN ESTRICTO, se pueden invertir)
5. Iniciar el análisis del esquema R para normalizarlo hasta BCNF (o 3FN). Comprobando si el esquema R se halla en BCNF.
    a. Si no está en BCNF, entonces intentar llevarlo a BCNF creando particiones del esquema R (ver **Proceso de particionamiento de un esquema analizando desde BCNF**)
    b. Si está en BCNF detener el proceso
6. Una vez finalizado el proceso escribir las particiones resultantes que se generan en el punto 3 (esquemas que están en BCNF o 3FN según corresponda)
7. Indicar la clave primaria resultante del proceso de normalización (tener en cuenta que debe haber quedado como clave primaria, alguna de las claves candidatas halladas en el punto 1)
8. Una vez identificadas las tablas que luego de aplicar el proceso de particionamiento de un esquema analizando desde BCNF quedaron en BCNF (o 3FN) analizar 4FN
    i. Si el proceso se realizó de acuerdo a lo propuesto en la materia, la única tabla que ***puede*** no estar en 4FN es la última que se halló con el proceso.
        a. En caso de detectar que un esquema no se encuentra en 4FN, ver el **Proceso de particionamiento de un esquema analizando 4FN**.
    ii. Una vez llevadas las particiones a 4FN, escribir de manera explícita al final de todo el proceso cuales son las tablas que considera que han quedado en 4FN y no son proyecciones de otras tablas.

Supongamos ahora que una vez aplicados los pasos 1 y 2 del proceso, al realizar el paso 3, se detecta un esquema que no se encuentra en BCNF. Dada está condición, se debe realizar el siguiente proceso de particionamiento:

Proceso de particionamiento de un esquema analizando desde BCNF

Cuando un esquema R no cumple la condición de BCNF, es decir existe en R al menos una dependencia funcional de la forma X→A tal que X no es superclave de R. Se debe proceder a la partición del esquema R. Para esto:

c) se genera una partición R_i conteniendo los atributos de la dependencia funcional X→A
d) se genera una partición R_{i+1} con los atributos de R – A. Está forma de particionar asegura que no se pierda información ya que:
   - $\mathbf{R_i \cap R_{i+1}}$ seguro es X por construcción, esto permite la relación entre los dos esquemas, y además
   - $\mathbf{R_i \cup R_{i+1} = R}$

¡Importante! En el caso de tener sobre R dependencias funcionales mínimas donde algunas de ellas son transitivas (al menos uno de los atributos del antecedente de una dependencia funcional es al menos uno de los atributos del consecuente de otra dependencia funcional), por ejemplo:

a,d-> b,e
b,f->c

se deberá tratar primero la dependencia funcional mínima cuyo antecedente contiene al menos uno de los atributos del consecuente de la otra dependencia funcional, en el ejemplo planteado, primero se debe tratar la dependencia funcional b,f->c y luego la dependencia funcional a,d-> b,e.
Si no se trata en el orden establecido previamente, la dependencia funcional se pierde por el orden en el que se tratan las mismas, sin que esto implique que el esquema no se pueda llevar a BCNF.

Una vez generadas las particiones se debe:

- Analizar la partición Ri indicando que dependencias funcionales mínimas de las indicadas sobre el esquema R valen en Ri. Indicar cual es la clave primaria del esquema Ri considerando la dependencia funcional mínima por la que se particionó el esquema original. Si las dependencias funcionales que quedaron valiendo en la partición Ri son mínimas, Ri quedo en BCNF.
  - **¡Importante!**
    - Si las dependencias funcionales planteadas sobre R no son mínimas, podría pasar que en Ri aparecieran nuevas dependencias funcionales que hagan que el esquema Ri no quedara en BCNF siendo necesario analizar BCNF en el esquema Ri. Para realizar este análisis es necesario reiniciar

el proceso de normalización desde el punto 3 ahora para el esquema R_i

- Si las dependencias funcionales halladas son mínimas pero no halle todas las claves candidatas del esquema y en consecuencia todas las dependencias funcionales válidas teniendo en cuenta esto, pueden surgir nuevas dependencias funcionales mínimas en R_i y seguir valiendo la definición de BCNF en R_i

- Analizar la partición R_{i+1} de modo que en ella queden las dependencias funcionales restantes aun no tratadas del esquema R. Es decir, aquellas dependencias funcionales que no quedaron siendo válidas en la partición R_i
  - Si están todas las dependencias funcionales, entonces marcar en la partición R_{i+1} las claves candidatas válidas en R y volver a analizar si ahora el esquema R_{i+1} se encuentra en BCNF. Para realizar este análisis es necesario reiniciar el proceso de normalización desde el punto 3 ahora para el esquema R_{i+1}
  - Si se detecta que hay atributos de una dependencia funcional que quedaron distribuidos en las particiones R_i y R_{i+1} analizar si se perdió la dependencia funcional usando el ***algoritmo de pérdida de dependencias funcionales*** para validarlo.
    - Si se perdió la dependencia funcional al particionar el esquema R para llevar a BCNF entonces, el esquema R no puede ser llevado a BCNF y se debe analizar 3 FN (en el caso de que BCNF falla y se debe analizar 3FN en el esquema R ver **Proceso de particionamiento de un esquema cuando falla BCNF y se debe analizar 3FN**). Luego de este análisis se pasa al punto 4 del proceso.
    - Si no se perdió ninguna dependencia funcional, se procede a analizar la partición R_{i+1}. En este punto, se vuelve a analizar si ahora el esquema R_{i+1} se encuentra en BCNF. Para realizar este análisis es necesario reiniciar el proceso de normalización desde el punto 3 ahora para el esquema R_{i+1}

      Aclaración: Si se detecta que la partición R_{i+1} contiene sólo el conjunto de atributos que conforman alguna de las claves candidatas halladas para el esquema R, se puede afirmar que el esquema R_{i+1} ya está en BCNF terminando así el proceso.

Una vez finalizado el paso 3 del proceso, se pasa al punto 4 continuando con el proceso hasta el punto 6.

Proceso de particionamiento de un esquema cuando falla BCNF y se debe analizar 3FN

A este punto de análisis se llega cuando desde el proceso de particionamiento de un esquema a BCNF, sucede que se perdió una dependencia funcional al particionar el esquema R en las

particiones R_i y R_{i+1}, entonces, el esquema R no puede ser llevado a BCNF y se debe analizar 3 FN sobre el esquema R.
Cuando esto sucede, existen dependencias funcionales que valen en el esquema original a normalizar y que aún no han sido tratadas y dejadas en tablas intermedias normalizadas a BCNF o 3 FN.

Para llevar el esquema R a 3FN

- *Si existe una única clave candidata válida en el esquema original a normalizar:*

  Se debe crear para cada una de estas dependencias funcionales válidas en R una R_i. En caso de que la clave candidata no haya quedado en ninguna de las particiones R_i se debe agregar una partición solo con el conjunto de atributos de la clave candidata.

- *Si existe más de una clave candidata en el esquema original a normalizar:*

  Al existir más de una clave candidata, hay dependencias funcionales que se corresponden con una clave candidata y otras que se corresponden con las restantes claves candidatas. Se toma una clave candidata como clave primaria, se contemplan las dependencias funcionales que se corresponden con está decisión y se particiona como en el caso anterior.

Proceso de particionamiento de un esquema analizando 4FN

A este punto del proceso se llega luego de realizar los pasos 1 al 5 del proceso de normalización, al realizar el paso 6 de dicho proceso, se detecta que el esquema **R** no se encuentra en 4FN, es decir existe al menos una dependencia multivaluada no trivial en **R**. Dada está condición, se deben realizar los siguientes pasos:

Plantear las dependencias multivaluadas válidas en **R** (considerar independencia de atributos analizadas a partir de las restricciones dadas para el esquema **R**). Si el esquema **R** no cumple la condición de 4FN, es decir existe en **R** al menos una dependencia multivaluada de la forma $X \rightarrow> Y$ que no es trivial en **R**. Se debe proceder a la partición del esquema **R**. Para esto:

- se genera una partición R_i conteniendo los atributos de la dependencia multivaluada $X \rightarrow> Y$
- se genera una partición R_{i+1} con los atributos de $R - Y$

Una vez generadas las particiones se debe:

- Indicar en la partición $\mathbf{R_i}$ la clave primaria, la cual contendrá todos los atributos del esquema $\mathbf{R_i}$.

- Indicar en la partición **R $_{i+1}$** la clave primaria, la cual contendrá todos los atributos del esquema **R $_{i+1}$**. Volver a analizar si ahora el esquema **R $_{i+1}$** se encuentra en 4FN. Para realizar este análisis es necesario reiniciar el **proceso de particionamiento de un esquema analizando 4FN** (puede ocurrir que surjan nuevas dependencias multivaluadas que son válidas en R $_{i+1}$ pero no eran válidas en R)

Ejemplos completos donde se usa el proceso de normalización, dado y evaluado en la materia

A continuación se da una serie de ejemplos en los que se aplica el proceso de normalización hasta 4FN

Ejemplo 1: Supongamos que tenemos el siguiente esquema *FIESTA*

Se retoma el ejemplo en el que se aplicó el proceso de normalización sobre el esquema **FIESTA**, sin que falle BCNF

FIESTA (#Salon, dirección, capacidad, fechaFiesta, nomContratante, cantInvitados, nombreInvitado, cantMesas, mesaInvitado, servicioContratado, dirContratante, dniInvitado)

Sobre el esquema **FIESTA** valen las siguientes restricciones:

- En cada salón se realiza una sola fiesta por día
- En un día puede haber varias fiestas en diferentes salones.
- Para cada fiesta puede figurar más de un contratante.
- En distintas fiestas el mismo contratante puede figurar con diferentes direcciones.
- Cada invitado tiene asociado un número de mesa.
- La cantidad de mesas del salón varía para cada fiesta.
- Servicio contratado es una lista (En la tabla aparecerá una tupla por cada servicio contratado) que describe los tipos de comida contratados para una fiesta.
- Una persona puede ir a más de una fiesta en el mismo salón.

Iniciamos el proceso de normalización:

Paso 1) Marcar en el esquema FIESTA la o las claves candidatas (CC)

CC: (#Salon, fechaFiesta, dniInvitado, nombreContratante, servicioContratado)

Paso 2) Escribir las dependencias funcionales no triviales y mínimas válidas en el esquema FIESTA

6. #Salon → dirección, capacidad
7. dniInvitado→ nombreInvitado
8. #Salon, fechaFiesta, nombreContratante → dirContratante
9. #Salon, fechaFiesta, dniInvitado → mesaInvitado
10. #Salon, fechaFiesta → cantInvitados, cantMesas

Paso 3) Iniciar el análisis del esquema FIESTA para normalizarlo hasta BCNF (o 3FN). Comprobando si el esquema FIESTA se halla en BCNF.

El esquema **FIESTA** no está en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 1-5 no es superclave en el esquema **FIESTA** (Por ejemplo: la dependencia funcional 1). Entonces divido la tabla utilizando la dependencia funcional 1 quedando lo siguiente:

F1(#Salon, dirección, capacidad)
F2(#Salon, fechaFiesta, nombreContratante, cantInvitados, nombreInvitado, cantMesas, mesaInvitado, servicioContratado, dirContratante, dniInvitado)

El esquema **F1** está en BCNF ya que el antecedente de la dependencia funcional 1 es superclave en **F1** y vale la dependencia funcional 1 la cual se usó en el proceso de división. El esquema **F2** no está en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 2-5 no es superclave en **F2** (Por ejemplo: la dependencia funcional 2). Entonces divido la tabla **F2** utilizando la dependencia funcional 2 quedando lo siguiente:

F3(dniInvitado, nombreInvitado)
F4(#Salon, fechaFiesta, nombreContratante, cantInvitados, cantMesas, mesaInvitado, servicioContratado, dirContratante, dniInvitado)

El esquema **F3** está en BCNF ya que el antecedente de la dependencia funcional 2 es superclave de **F3** y vale la dependencia funcional 2 la cual se usó en el proceso de división. El esquema **F4** no está en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 3-5 no es superclave de **F4** (Por ejemplo: la dependencia funcional 3). Entonces divido la tabla **F4** utilizando la dependencia funcional 3 quedando lo siguiente:

F5(#Salon, fechaFiesta, nombreContratante, dirContratante)
F6(#Salon, fechaFiesta, nombreContratante, cantInvitados, cantMesas, mesaInvitado, servicioContratado, dniInvitado)

El esquema **F5** está en BCNF ya que el antecedente de la dependencia funcional 3 es superclave de **F5** y vale la dependencia funcional 3 la cual se usó en el proceso de división. El esquema **F6** no está en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 4 y 5 no es superclave de **F6** (Por ejemplo: la dependencia funcional 4). Entonces divido la tabla **F6** utilizando la dependencia funcional 4 quedando lo siguiente:

F7(#Salon, fechaFiesta, dniInvitado, mesaInvitado)
F8(#Salon,fechaFiesta,nombreContratante,cantInvitados,cantMesas,servicioContratado, dniInvitado)

El esquema **F7** está en BCNF ya que el antecedente de la dependencia funcional 4 es superclave de **F7** y vale la dependencia funcional 4 la cual se usó en el proceso de división. El esquema **F8** no está en BCNF porque el antecedente de la dependencia funcional 5 no

es superclave de **F8**. Entonces divido la tabla **F8** utilizando la dependencia funcional 5 quedando lo siguiente:

F9(#Salon, fechaFiesta,cantInvitados, cantMesas)
F10(#Salon fechaFiesta,nombreContratante, servicioContratado, dniInvitado)

El esquema **F9** está en BCNF ya que el antecedente de la dependencia funcional 5 es superclave de **F9** y vale la dependencia funcional 5 la cual se usó en el proceso de división. El esquema **F10** está en BCNF porque todos los atributos forman parte de la clave de dicho esquema.

Paso 4) Una vez finalizado el proceso escribir las particiones resultantes que se generan en el punto 3 (esquemas que están en BCNF o 3FN según corresponda)

Tablas en BCNF: F1, F3, F5, F7, F9 y F10.

Paso 5) Indicar la clave primaria resultante del proceso de normalización (tener en cuenta que debe haber quedado como clave primaria, alguna de las claves candidatas halladas en el punto 1)

Clave Primaria:(#Salon,fechaFiesta,dniInvitado,nombreContratante, servicioContratado)

Paso 6) Una vez identificadas las tablas que luego de aplicar el proceso de particionamiento de un esquema analizando desde BCNF quedaron en BCNF (o 3FN) analizar 4FN

Dependencias multivaluadas válidas en *F10* contemplando las restricciones dadas sobre el esquema:

1. #Salon, fechaFiesta -->> servicioContratado
2. #Salon, fechaFiesta -->> nombreContratante
3. #Salon, fechaFiesta -->> dniInvitado

El esquema **F10** no está en 4FN porque existen dependencias multivaluadas 1-3 que no son triviales en **F10**. Entonces divido la tabla **F10** utilizando la dependencia multivaluada 1 quedando lo siguiente:

F11(#Salon, fechaFiesta, servicioContratado)
F12(#Salon, fechaFiesta, nombreContratante, dniInvitado)

El esquema **F11** está en 4FN ya que sólo vale la dependencia multivaluada 1 que es trivial en dicho esquema.
El esquema **F12** no está en 4FN porque existen las dependencias multivaluadas 2 y 3 y éstas no son triviales en **F12**. Entonces divido la tabla **F12** utilizando la dependencia multivaluada 2 quedando lo siguiente:

F13(#Salon, fechaFiesta, nombreContratante)
F14(#Salon, fechaFiesta, dniInvitado)

El esquema **F13** está en 4FN ya que solo vale la dependencia multivaluada 2 que es trivial en dicho esquema.
El esquema **F14** está en 4FN ya que solo vale la dependencia multivaluada 3 que es trivial en dicho esquema.

Las tablas **F13** y **F14** son proyecciones de las tablas **F5** y **F7** respectivamente.

Nota: como se siguió el proceso propuesto, podemos asegurar que las tablas **F1, F3, F5, F7, F9** se encuentran en 4FN ya que no tiene dependencias multivaluadas.

Paso 6.1) Una vez llevadas las particiones a 4FN, escribir de manera explicita al final de todo el proceso cuales son las tablas que considera que han quedado en 4FN y no son proyecciones de otras tablas.

Tablas en 4FN: F1, F3, F5, F7, F9 y F11.

Ejemplo 2: Supongamos que tenemos el siguiente esquema INFORMACION_PAIS

INFORMACION_PAIS(#Pais, #Prov, atractivoTuristico)

Sobre el esquema **INFORMACION_PAIS** valen las siguientes restricciones:

- Un *#Pais* tiene varias provincias
- Una provincia puede estar en diferentes países
- Los atractivos turísticos son todos los atractivos de un país. El mismo atractivo puede estar en más de un país.

Iniciamos el proceso de normalización:

Paso 1) Marcar en el esquema INFORMACION_PAIS la o las claves candidatas (CC)

CC: (#Pais, #Prov, atractivoTuristico)

Paso 2) Escribir las dependencias funcionales no triviales y mínimas válidas en el esquema INFORMACION_PAIS

No hay dependencias funcionales.

Paso 3) Iniciar el análisis del esquema INFORMACION_PAIS para normalizarlo hasta BCNF (o 3FN). Comprobando si el esquema INFORMACION_PAIS se halla en BCNF.

El esquema INFORMACION_PAIS se encuentra en BCNF, ya que todos los atributos del esquema forman parte de la clave candidata.

Paso 4) Una vez finalizado el proceso escribir las particiones resultantes que se generan en el punto 3 (esquemas que están en BCNF o 3FN según corresponda)

Tablas en BCNF: INFORMACION_PAIS

Paso 5) Indicar la clave primaria resultante del proceso de normalización (tener en cuenta que debe haber quedado como clave primaria, alguna de las claves candidatas halladas en el punto 1)

Clave Primaria: (#Pais, #Prov, atractivoTuristico)

Paso 6) Una vez identificadas las tablas que luego de aplicar el proceso de particionamiento de un esquema analizando desde BCNF quedaron en BCNF (o 3FN) analizar 4FN

Dependencias multivaluadas válidas en *INFORMACION_PAIS* contemplando las restricciones dadas sobre el esquema:

1. #Pais -->> #Prov
2. #Pais -->> atractivoTurístico

El esquema **INFORMACION_PAIS** no está en 4FN porque existen las dependencias multivaluadas 1 y 2 que no son triviales en **INFORMACION_PAIS**. Entonces divido la tabla **INFORMACION_PAIS** utilizando la dependencia multivaluada 1 quedando lo siguiente:

I1 (#Pais, #Prov)
I2 (#Pais, atractivoTurístico)

El esquema **I1** está en 4FN ya que sólo vale la dependencia multivaluada 1 que es trivial en dicho esquema.
El esquema **I2** está en 4FN ya que sólo vale la dependencia multivaluada 2 que es trivial en dicho esquema.

Paso 6.1) Una vez llevadas las particiones a 4FN, escribir de manera explicita al final de todo el proceso cuales son las tablas que considera que han quedado en 4FN y no son proyecciones de otras tablas.

Tablas en 4FN: **I1**, **I2**

Ejemplo 3: Supongamos que tenemos el siguiente esquema INFORMACION_CURSO

INFORMACION_CURSO (#Curso,material,profesor)

Sobre el esquema **INFORMACION_CURSO** valen las siguientes restricciones:

- Los cursos tienen varios profesores asignados
- Un curso tiene asignado muchos materiales. Los materiales son propios del curso. Cuando un profesor dicta un curso ya los tiene disponibles

Iniciamos el proceso de normalización:

Paso 1) Marcar en el esquema INFORMACION_CURSO la o las claves candidatas (CC)

CC: (#Curso,material,profesor)

Paso 2) Escribir las dependencias funcionales no triviales y mínimas válidas en el esquema INFORMACION_CURSO

No hay dependencias funcionales.

Paso 3) Iniciar el análisis del esquema INFORMACION_CURSO para normalizarlo hasta BCNF (o 3FN). Comprobando si el esquema INFORMACION_CURSO se halla en BCNF.

El esquema INFORMACION_CURSO se encuentra en BCNF, ya que todos los atributos forman parte de la clave candidata.

Paso 4) Una vez finalizado el proceso escribir las particiones resultantes que se generan en el punto 3 (esquemas que están en BCNF o 3FN según corresponda)

Tablas en BCNF: INFORMACION_CURSO

Paso 5) Indicar la clave primaria resultante del proceso de normalización (tener en cuenta que debe haber quedado como clave primaria, alguna de las claves candidatas halladas en el punto 1)

Clave Primaria: (#Curso,material,profesor)

Paso 6) Una vez identificadas las tablas que luego de aplicar el proceso de particionamiento de un esquema analizando desde BCNF quedaron en BCNF (o 3FN) analizar 4FN

Dependencias multivaluadas válidas en INFORMACION_CURSO contemplando las restricciones dadas sobre el esquema:

1. #Curso -->> material
2. #Curso -->> profesor

El esquema **INFORMACION_CURSO** no está en 4FN porque existen las dependencias multivaluadas 1 y 2 que no son triviales en **INFORMACION_CURSO**. Entonces divido la tabla **INFORMACION_CURSO** utilizando la dependencia multivaluada 1 quedando lo siguiente:

INFO1 (#Curso,material)
INFO2 (#Curso,profesor)

El esquema **INFO1** está en 4FN ya que sólo vale la dependencia multivaluada 1 que es trivial en dicho esquema.
El esquema **INFO2** está en 4FN ya que sólo vale la dependencia multivaluada 2 que es trivial en dicho esquema.

Paso 6.1) Una vez llevadas las particiones a 4FN, escribir de manera explicita al final de todo el proceso cuales son las tablas que considera que han quedado en 4FN y no son proyecciones de otras tablas.

Tablas en 4FN: **INFO1**, **INFO2**

Ejemplo 4: Supongamos que tenemos el siguiente esquema CURSO

CURSO (#Curso,material,profesor)

Sobre el esquema **CURSO** valen las siguientes restricciones:

- Los cursos tienen varios profesores asignados
- cada profesor dentro de un curso usa distintos materiales didácticos.

Iniciamos el proceso de normalización:

Paso 1) Marcar en el esquema CURSO la o las claves candidatas (CC)

CC: (#Curso,material,profesor)

Paso 2) Escribir las dependencias funcionales no triviales y mínimas válidas en el esquema CURSO

No hay dependencias funcionales.

Paso 3) Iniciar el análisis del esquema CURSO para normalizarlo hasta BCNF (o 3FN). Comprobando si el esquema CURSO se halla en BCNF.

El esquema CURSO se encuentra en BCNF, ya que todos los atributos forman parte de la clave candidata.

Paso 4) Una vez finalizado el proceso escribir las particiones resultantes que se generan en el punto 3 (esquemas que están en BCNF o 3FN según corresponda)

Tablas en BCNF: CURSO

Paso 5) Indicar la clave primaria resultante del proceso de normalización (tener en cuenta que debe haber quedado como clave primaria, alguna de las claves candidatas halladas en el punto 1)

Clave Primaria: (#Curso,material,profesor)

Paso 6) Una vez identificadas las tablas que luego de aplicar el proceso de particionamiento de un esquema analizando desde BCNF quedaron en BCNF (o 3FN) analizar 4FN

Dependencias multivaluadas válidas en CURSO contemplando las restricciones dadas sobre el esquema:

1. #Curso, profesor -->> material

El esquema **CURSO** está en 4FN porque existe solo la dependencia multivaluada 1, la cual es trivial en el esquema **CURSO**.

Paso 6.1) Una vez llevadas las particiones a 4FN, escribir de manera explicita al final de todo el proceso cuales son las tablas que considera que han quedado en 4FN y no son proyecciones de otras tablas.

Tablas en 4FN: **CURSO**

Ejemplo 5: Supongamos que tenemos el siguiente esquema VIDEO_CLUB

VIDEO_CLUB (#Cliente, #Video)

Sobre el esquema **VIDEO_CLUB** valen las siguientes restricciones:

- #Cliente representa a todos los clientes del video club
- #Video representa a cada uno de los videos existentes en el video club

Iniciamos el proceso de normalización:

Paso 1) Marcar en el esquema VIDEO_CLUB la o las claves candidatas (CC)

CC:(#Cliente, #Video)

Paso 2) Escribir las dependencias funcionales no triviales y mínimas válidas en el esquema VIDEO_CLUB

No hay dependencias funcionales.

Paso 3) Iniciar el análisis del esquema VIDEO_CLUB para normalizarlo hasta BCNF (o 3FN). Comprobando si el esquema VIDEO_CLUB se halla en BCNF.

El esquema VIDEO_CLUB se encuentra en BCNF, ya que todos los atributos forman parte de la clave candidata.

Paso 4) Una vez finalizado el proceso escribir las particiones resultantes que se generan en el punto 3 (esquemas que están en BCNF o 3FN según corresponda)

Tablas en BCNF: VIDEO_CLUB

Paso 5) Indicar la clave primaria resultante del proceso de normalización (tener en cuenta que debe haber quedado como clave primaria, alguna de las claves candidatas halladas en el punto 1)

Clave Primaria: (#Cliente, #Video)

Paso 6) Una vez identificadas las tablas que luego de aplicar el proceso de particionamiento de un esquema analizando desde BCNF quedaron en BCNF (o 3FN) analizar 4FN

Dependencias multivaluadas válidas en VIDEO_CLUB contemplando las restricciones dadas sobre el esquema:

1. ∅ -->> #Cliente
2. ∅ -->> #Video

El esquema **VIDEO_CLUB** no está en 4FN porque existen dependencias multivaluadas 1 y 2 que no son triviales en **VIDEO_CLUB**. Entonces divido la tabla **VIDEO_CLUB** utilizando la dependencia multivaluada 1 quedando lo siguiente:

V1(#Cliente)
V2(#Video)

El esquema **V1** está en 4FN ya que sólo vale la dependencia multivaluada 1 que es trivial en dicho esquema.
El esquema **V2** está en 4FN ya que sólo vale la dependencia multivaluada 2 que es trivial en dicho esquema.

Paso 6.1) Una vez llevadas las particiones a 4FN, escribir de manera explicita al final de todo el proceso cuales son las tablas que considera que han quedado en 4FN y no son proyecciones de otras tablas.

Tablas en 4FN: **V1**, **V2**

Ejemplo 6: Supongamos que tenemos el siguiente esquema UNIDAD_ACADEMICA que representa la información de una unidad académica

UNIDAD_ACADEMICA (#Curso,material,profesor, #Alumno)

Sobre el esquema **UNIDAD_ACADEMICA** valen las siguientes restricciones:

- Los cursos tienen varios profesores asignados.
- Cada profesor dentro de un curso usa distintos materiales didácticos.
- #Alumno representa a cada uno de los alumnos de la unidad académica.

Iniciamos el proceso de normalización:

Paso 1) Marcar en el esquema UNIDAD_ACADEMICA la o las claves candidatas (CC)

CC: (#Curso,material,profesor, #Alumno)

Paso 2) Escribir las dependencias funcionales no triviales y mínimas válidas en el esquema UNIDAD_ACADEMICA

No hay dependencias funcionales.

Paso 3) Iniciar el análisis del esquema UNIDAD_ACADEMICA para normalizarlo hasta BCNF (o 3FN). Comprobando si el esquema UNIDAD_ACADEMICA se halla en BCNF.

El esquema UNIDAD_ACADEMICA se encuentra en BCNF, ya que todos los atributos forman parte de la clave candidata.

Paso 4) Una vez finalizado el proceso escribir las particiones resultantes que se generan en el punto 3 (esquemas que están en BCNF o 3FN según corresponda)

Tablas en BCNF: UNIDAD_ACADEMICA

Paso 5) Indicar la clave primaria resultante del proceso de normalización (tener en cuenta que debe haber quedado como clave primaria, alguna de las claves candidatas halladas en el punto 1)

Clave Primaria: (#Curso,material,profesor, #Alumno)

Paso 6) Una vez identificadas las tablas que luego de aplicar el proceso de particionamiento de un esquema analizando desde BCNF quedaron en BCNF (o 3FN) analizar 4FN

Dependencias multivaluadas válidas en UNIDAD_ACADEMICA contemplando las restricciones dadas sobre el esquema:

1. #Curso, profesor -->> material
2. ∅ -->> #Alumno

El esquema **UNIDAD_ACADEMICA** no está en 4FN porque existen las dependencias multivaluadas 1 y 2 que no son triviales en **UNIDAD_ACADEMICA**. Entonces divido la tabla **UNIDAD_ACADEMICA** utilizando la dependencia multivaluada 1 quedando lo siguiente:

U1 (#Curso, profesor ,material)
U2 (#Curso,profesor, #Alumno)

El esquema **U1** está en 4FN ya que sólo vale la dependencia multivaluada 1 que es trivial en dicho esquema.
En el esquema **U2** ahora sigue valiendo la dependencia multivaluada 2 (∅ -->> #Alumno) y empieza a valer la siguiente dependencia multivaluada en el esquema **U2**:

3. #Curso -->> profesor

El esquema **U2** no está en 4FN porque existen las dependencias multivaluadas 2 y 3 que no son triviales en **U2**. Entonces divido la tabla **U2** utilizando la dependencia multivaluada 3 (notar acá que el orden en el que se tratan las multivaluadas, puede hacer que en la partición resultante aparezcan nuevas dependencias multivaluadas) quedando lo siguiente:

U3 (#Curso,profesor)
U4 (#Curso, #Alumno)

El esquema **U3** está en 4FN ya que sólo vale la dependencia multivaluada 3 que es trivial en dicho esquema.
En el esquema **U4** ahora sigue valiendo la dependencia multivaluada 2 (∅ **-->>** #Alumno) y empieza a valer la siguiente dependencia multivaluada en el esquema **U4**:

4. ∅ **-->>** #Curso

El esquema **U4** no está en 4FN porque existen las dependencias multivaluadas 2 y 4 que no son triviales en **U4**. Entonces divido la tabla **U4** utilizando la dependencia multivaluada 4 quedando lo siguiente:

U5 (#Curso)
U6 (#Alumno)

El esquema **U5** está en 4FN ya que sólo vale la dependencia multivaluada 4 que es trivial en dicho esquema.
El esquema **U6** está en 4FN ya que sólo vale la dependencia multivaluada 2 que es trivial en dicho esquema.

Las tablas **U3** y **U5** son proyecciones de la tabla **U1**.

Paso 6.1) Una vez llevadas las particiones a 4FN, escribir de manera explicita al final de todo el proceso cuales son las tablas que considera que han quedado en 4FN y no son proyecciones de otras tablas.

Tablas en 4FN: **U1**, **U6**

Ejemplo 7: Supongamos que tenemos el siguiente esquema UNIDAD_EDUCATIVA que representa la información de una unidad educativa

UNIDAD_EDUCATIVA (#Curso,material,profesor, #Alumno)

Sobre el esquema **UNIDAD_EDUCATIVA** valen las siguientes restricciones:

- Los cursos tienen varios profesores asignados.
- cada profesor dentro de un curso usa distintos materiales didácticos.
- #Alumno representa a cada uno de los alumnos de la unidad educativa.

Importante:

La única diferencia con el ejemplo 6 se da en el paso 6 del proceso de normalización en el que las dependencias multivaluadas se tratan en otro orden lo que modifica el análisis y proceso de 4FN. Como resultado del proceso tanto del ejemplo 6 como del ejemplo 7 las tablas resultantes son las mismas.

Iniciamos el proceso de normalización:

Paso 1) Marcar en el esquema UNIDAD_EDUCATIVA la o las claves candidatas (CC)

CC: (#Curso,material profesor, #Alumno)

Paso 2) Escribir las dependencias funcionales no triviales y mínimas válidas en el esquema UNIDAD_EDUCATIVA

No hay dependencias funcionales.

Paso 3) Iniciar el análisis del esquema UNIDAD_EDUCATIVA para normalizarlo hasta BCNF (o 3FN). Comprobando si el esquema UNIDAD_EDUCATIVA se halla en BCNF.

El esquema UNIDAD_EDUCATIVA se encuentra en BCNF, ya que todos los atributos forman parte de la clave candidata.

Paso 4) Una vez finalizado el proceso escribir las particiones resultantes que se generan en el punto 3 (esquemas que están en BCNF o 3FN según corresponda)

Tablas en BCNF: UNIDAD_EDUCATIVA

Paso 5) Indicar la clave primaria resultante del proceso de normalización (tener en cuenta que debe haber quedado como clave primaria, alguna de las claves candidatas halladas en el punto 1)

Clave Primaria: (#Curso,material,profesor, #Alumno)

Paso 6) Una vez identificadas las tablas que luego de aplicar el proceso de particionamiento de un esquema analizando desde BCNF quedaron en BCNF (o 3FN) analizar 4FN

Dependencias multivaluadas válidas en UNIDAD_EDUCATIVA contemplando las restricciones dadas sobre el esquema:

1. #Curso, profesor -->> material
2. ∅ -->> #Alumno

El esquema **UNIDAD_EDUCATIVA** no está en 4FN porque existen las dependencias multivaluadas 1 y 2 que no son triviales en **UNIDAD_EDUCATIVA**. Entonces divido la tabla **UNIDAD_EDUCATIVA** utilizando la dependencia multivaluada 2 quedando lo siguiente:

E1 (#Alumno)
E2 (#Curso, profesor ,material)

El esquema **E1** está en 4FN ya que sólo vale la dependencia multivaluada 2 que es trivial en dicho esquema.
El esquema **E2** está en 4FN ya que sólo vale la dependencia multivaluada 1 que es trivial en dicho esquema.

Paso 6.1) Una vez llevadas las particiones a 4FN, escribir de manera explicita al final de todo el proceso cuales son las tablas que considera que han quedado en 4FN y no son proyecciones de otras tablas.

Tablas en 4FN: **E1**, **E2**

Ejemplo 8: Supongamos que tenemos el siguiente esquema PROFESOR_CURSO que representa la información de una unidad académica

PROFESOR_CURSO(#Curso,material,profesor)

Sobre el esquema **PROFESOR_CURSO** valen las siguientes restricciones:

- Los cursos tienen varios profesores asignados.
- Un profesor usa materiales independientemente del curso. Esto quiere decir que el material es usado por cada profesor en cada curso que dicta

Iniciamos el proceso de normalización:

Paso 1) Marcar en el esquema PROFESOR_CURSO la o las claves candidatas (CC)

CC: (#Curso,material profesor)

Paso 2) Escribir las dependencias funcionales no triviales y mínimas válidas en el esquema PROFESOR_CURSO

No hay dependencias funcionales.

Paso 3) Iniciar el análisis del esquema PROFESOR_CURSO para normalizarlo hasta BCNF (o 3FN). Comprobando si el esquema PROFESOR_CURSO se halla en BCNF.

El esquema PROFESOR_CURSO se encuentra en BCNF, ya que todos los atributos forman parte de la clave candidata.

Paso 4) Una vez finalizado el proceso escribir las particiones resultantes que se generan en el punto 3 (esquemas que están en BCNF o 3FN según corresponda)

Tablas en BCNF: PROFESOR_CURSO

Paso 5) Indicar la clave primaria resultante del proceso de normalización (tener en cuenta que debe haber quedado como clave primaria, alguna de las claves candidatas halladas en el punto 1)

Clave Primaria: (#Curso,material,profesor)

Paso 6) Una vez identificadas las tablas que luego de aplicar el proceso de particionamiento de un esquema analizando desde BCNF quedaron en BCNF (o 3FN) analizar 4FN

Dependencias multivaluadas válidas en PROFESOR_CURSO contemplando las restricciones dadas sobre el esquema:

1. profesor -->> material

¡Atención! Algo que podría surgir en este punto del análisis es la idea de proponer las siguientes dependencias multivaluadas

profesor -->> material
#Curso ->> profesor

Como se vio en la parte de análisis de dependencias multivaluadas anteriormente en el apunte, la dependencia multivaluada

#Curso ->> profesor no es válida

ya que a está altura no hay independencia del atributo profesor en relación al atributo material.

Aquí lo que pasa es que se debe plantear la dependencia multivaluada

profesor -->> material

sobre el esquema **PROFESOR_CURSO** para que luego de la partición valga la dependencia multivaluada #Curso ->> profesor

El esquema **PROFESOR_CURSO** no está en 4FN porque existe la dependencia multivaluada 1 que no es trivial en **PROFESOR_CURSO**. Entonces divido la tabla **PROFESOR_CURSO** utilizando la dependencia multivaluada 1 quedando lo siguiente:

P1 (profesor ,material)
P2 (#Curso, profesor)

El esquema **P1** está en 4FN ya que sólo vale la dependencia multivaluada 1 que es trivial en dicho esquema.
En el esquema **P2** ahora empieza a valer la siguiente dependencia multivaluada en el esquema **P2**:

2. #Curso -->> profesor

El esquema **P2** está en 4FN ya que sólo vale la dependencia multivaluada 2 que es trivial en dicho esquema.

Paso 6.1) Una vez llevadas las particiones a 4FN, escribir de manera explicita al final de todo el proceso cuales son las tablas que considera que han quedado en 4FN y no son proyecciones de otras tablas.

Tablas en 4FN: **P1**, **P2**

Ejemplo 9: Supongamos que tenemos el siguiente esquema R

Se retoma el ejemplo en el que se aplicó el proceso de normalización sobre el esquema **R**, y por las características halladas, se debió aplicar el Proceso de particionamiento de un esquema cuando falla BCNF y se debió analizar 3FN

R(a,b,c,d,e,f,g)

Paso 1) Marcar en el esquema R la o las claves candidatas(CC)

CC: (d,e,a)

Paso 2) Escribir las dependencias funcionales no triviales y mínimas válidas en el esquema R

5. a → b,c
6. d,e → f
7. e → g
8. a → g

Paso 3) Iniciar el análisis del esquema R para normalizarlo hasta BCNF (o 3FN). Comprobando si el esquema R se halla en BCNF.

El esquema **R** no está en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 1-4 no es superclave de **R** (Por ejemplo: la dependencia funcional 1). Entonces divido la tabla utilizando la dependencia funcional 1 quedando lo siguiente:

R1(**a**,b,c)
R2(**d,e**,f,g,**a**)

El esquema **R1** está en BCNF ya que el antecedente de la dependencia funcional 1 es superclave de **R1** y vale la dependencia funcional 1 la cual se usó en el proceso de división. El esquema **R2** no está en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 2-4 no es superclave de **R2** (Por ejemplo: la dependencia funcional 2). Entonces divido la tabla **R2** utilizando la dependencia funcional 2 quedando lo siguiente:

R3(**d,e**,f)
R4(**d,e,a**,g)

El esquema **R3** está en BCNF ya que el antecedente de la dependencia funcional 2 es superclave en **R3** y vale la dependencia funcional 2 la cual se usó en el proceso de división. El esquema **R4** no está en BCNF porque al menos uno de los antecedentes de las dependencias funcionales 3 y 4 no es superclave de **R4** (Por ejemplo: la dependencia funcional 4). Entonces divido la tabla **R4** utilizando la dependencia funcional 4 quedando lo siguiente:

R5(**a**,g)
R6(**a,d,e**)

- Se debe analizar si se perdió la dependencia funcional e → g

Para esto aplico el algoritmo para determinar pérdida de dependencias funcionales

RES: (e)
Mientras Res cambia
Para i= 1 to cant_de_ particiones_realizadas
Res = Res ∪ ((Res ∩ Ri)$^+$ ∩ Ri)

Desde i = 1 hasta 4 (Son las particiones R1-R3-R5-R6)

Paso 1)

R1(a,b,c)
Res = (e) ∪ (((e) ∩ (a,b,c))$^+$ ∩ (a,b,c))
Res = (e)

Paso 2)

R3(d,e,f)
Res = (e) ∪ (((e) ∩ (d,e,f))$^+$ ∩ (d,e,f))
Res = (e) ∪ ((e)$^+$ ∩ (d,e,f))

-Debo hallar la clausura del conjunto de atributos (e)
Result:= X
While (hay cambios en result) do
For (cada dependencia funcional Y → Z en F) do
if (Y ⊆ result) then
result := result ∪ Z

result : (e)
para cada dependencia funcional del conjunto
{ a → b,c
d,e → f
e → g
a → g}
Entra al while por primera vez y result= (e,g)

Como result cambio, entro una vez mas al while
result : (e,g)
para cada dependencia funcional del conjunto
{ a → b,c
d,e → f
e → g
a → g }
En está iteración, result queda igual y termino la iteración
result : (e,g)

Res = (e) ∪ ((e,g) ∩ (d,e,f))
Res = (e) ∪ (e)
Res = (e)

Paso 3)

R5(a,g)
Res = (e) ∪ (((e) ∩ (a,g))$^+$ ∩ (a,g))
Res = (e)

Paso 4)

R6(a,d,e)
Res = (e) ∪ (((e) ∩ (a,d,e))$^+$ ∩ (a,d,e))
Res = (e) ∪ ((e)$^+$ ∩ (a,d,e))

-Debo hallar la clausura del conjunto de atributos (e)
Result:= X
While (hay cambios en result) do
For (cada dependencia funcional Y → Z en F) do
if (Y ⊆ result) then
result := result ∪ Z

result : (e)
para cada dependencia funcional del conjunto
{ a → b,c
d,e → f
e → g
a → g}
Entra al while por primera vez y result= (e,g)

Como result cambio, entro una vez mas al while
result : (e,g)
para cada dependencia funcional del conjunto
{ a → b,c
d,e → f
e → g
a → g }
En está iteración, result queda igual y termino la iteración
result : (e,g)

Res = (e) ∪ ((e,g) ∩ (a,d,e))
Res = (e) ∪ (e)
Res = (e)

Con la aplicación del algoritmo de pérdida de dependencias funcionales, se demuestra que con está división se pierde la dependencia funcional e → g, entonces hay aplicar el proceso de particionamiento de un esquema cuando falla BCNF analizando 3FN.
Dado que sólo existe una clave candidata, se arma una tabla con cada una de las dependencias funcionales que quedan y otra con la clave.

R4.1(**e**,g)
R4.2(**a**,g)
R4.3(**d,e,a**)

Paso 4) Una vez finalizado el proceso escribir las particiones resultantes que se generan en el punto 3 (esquemas que están en BCNF o 3FN según corresponda)

Tablas en BCNF: R1, R3
Tablas en 3FN: R4.1, R4.2 y R4.3

Paso 5) Indicar la clave primaria resultante del proceso de normalización (tener en cuenta que debe haber quedado como clave primaria, alguna de las claves candidatas halladas en el punto 1)

Clave Primaria: (d,e,a)

Paso 6) Una vez identificadas las tablas que luego de aplicar el proceso de particionamiento de un esquema analizando desde BCNF quedaron en BCNF (o 3FN) analizar 4FN

Al tratarse de un ejemplo abstracto, trabajaremos sobre los siguientes supuestos:

- *Dado un valor para el atributo d, existen muchos valores del atributo e asociados.*
- *El atributo a toma todos los valores posibles para ese atributo independientemente de los atributos d y e.*

Bajo estos supuestos, decimos que las dependencias multivaluadas válidas en R4.3 son:

1. ∅ -->> a
2. d -->> e

El esquema **R4.3** no está en 4FN porque existen las dependencias multivaluadas 1 y 2 que no son triviales en **R4.3**. Entonces divido la tabla **R4.3** utilizando la dependencia multivaluada 1 quedando lo siguiente:

R4.3.1(a)
R4.3.2(d,e)

El esquema **R4.3.1** está en 4FN ya que sólo vale la dependencia multivaluada 1 que es trivial en dicho esquema.
El esquema **R4.3.2** está en 4FN ya que sólo vale la dependencia multivaluada 2 que es trivial en dicho esquema.

Las tablas **R4.3.1** y **R4.3.2** son proyecciones de las tablas **R1** y **R3** respectivamente.

Nota: como se siguió el proceso propuesto, podemos asegurar que las tablas **R1, R3, R4.1, R4.2** se encuentran en 4FN ya que no tienen dependencias multivaluadas.

Paso 6.1) Una vez llevadas las particiones a 4FN, escribir de manera explicita al final de todo el proceso cuales son las tablas que considera que han quedado en 4FN y no son proyecciones de otras tablas.

Tablas en 4FN: **R1,R3, R4.1, R4.2**