



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE CIENCIAS
FUNDAMENTOS DE BASES DE DATOS

Normalización de Bases de Datos

Gerardo Avilés Rosas
gar@ciencias.unam.mx

Aspectos de diseño de BD

Uno de los principales problemas que se presentan cuando se convierten directamente diseños de BDs del modelo E/R → Modelo Relacional es la **redundancia**.

La **redundancia** consiste en que un *hecho* se repita en más de una **tupla**.

Una de las causas más común en redundancia es intentar agrupar en una relación las propiedades **multivaluadas** y **univaluadas** de un objeto.



...Aspectos de diseño de BD

Vamos a suponer que tenemos la siguiente relación:

nombre_suc	delegación	activo	nom_cliente	nPréstamo	importe
Centro	Cuauhtémoc	1,800 M	Santos	P – 17	200,000
Copilco	Coyoacán	420 M	Gómez	P – 23	400,000
Viveros	Coyoacán	340 M	López	P – 15	300,000
Centro	Cuauhtémoc	1,800 M	Toledo	P – 14	300,000
Eugenia	Benito Juárez	80 M	Santos	P – 93	100,000
Zapata	Benito Juárez	1,600 M	Abril	P – 11	180,000
San Ángel	Álvaro	60 M	Vázquez	P – 29	240,000
Tlalpan	Obregón	740 M	López	P – 16	260,000
Centro	Tlalpan	1,800 M	González	P – 23	400,000
Viveros	Coyoacán	340 M	Rodríguez	P – 25	500,000

¿Existe algún tipo de redundancia?

...Aspectos de diseño de BD

- **Redundancia de información:**

agregar un nuevo préstamo o cambiar la dirección de una sucursal (duplicar información).

- **Incapacidad para representar cierta información:**

¿Cómo dar de alta una nueva sucursal?
¿Qué sucede si no hubiera préstamos?

Es decir, existen **anomalías** en la base de datos.

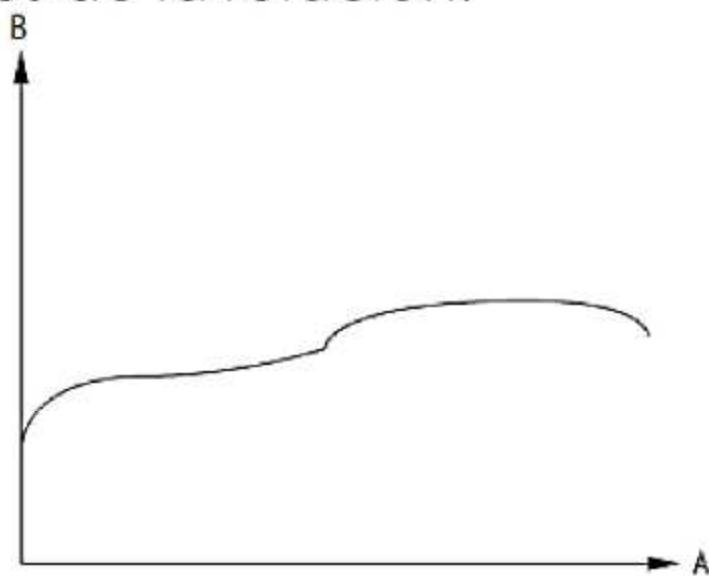
Una **anomalía** es un problema que surge en una Base de Datos. Las principales anomalías que se pueden encontrar son:

- **Redundancia:** La información puede repetirse innecesariamente en varias *tuplas*.
- **Anomalías de actualización:** Podemos cambiar información en una *tupla* y dejarla inalterada en otra.
- **Anomalías de eliminación:** Si un conjunto de valores queda vacío, podemos perder información adicional como efecto secundario.

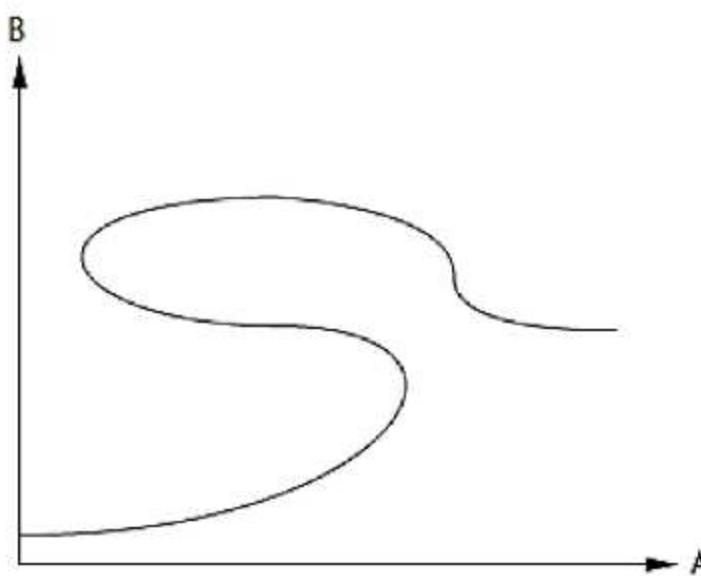


Dependencias funcionales

Las **dependencias funcionales** ayudan a especificar formalmente cuándo un diseño es correcto. Se trata de una relación unidireccional entre 2 atributos de tal forma que en un momento dado, para cada valor único de A, sólo un valor de B se asocia con él a través de la relación.



A functionally determines B. Each value of A corresponds to only one value of B.



A does not functionally determine B. Some values of A correspond to more than one value of B.

Una **DF** que denotaremos por $X \rightarrow Y$, sucede entre dos conjuntos de atributos X e Y que son subconjuntos de R.

...Dependencias funcionales

Por ejemplo, vamos a suponer que tenemos la relación:

Pedidos (*idCliente, nombre, paterno, materno, calle, ciudad, estado, CP, teléfono, noPedido fechaOrden, noArticulo, precio, ¿enviado?*)

Se espera, por ejemplo que:

idCliente → *nombre, paterno, materno, calle, ciudad, estado, CP, teléfono*

noPedido → *fechaOrden, Precio, ¿Enviado?*

Aunque los valores de los atributos pueden cambiar, en cualquier momento, sólo hay uno.

Las DFs se utilizan para:

- **Especificar restricciones sobre el conjunto de relaciones.**
- **Examinar las relaciones y determinar si son legales bajo un conjunto de dependencias funcionales dado.**

...Dependencias funcionales

- Si X es una llave de R esto implica que $X \rightarrow Y$ para cualquier subconjunto de atributos Y de R.
- Si $X \rightarrow Y$ en R, esto no implica que $Y \rightarrow X$
- Una **DF** es una **propiedad semántica** de los atributos, la cual se debe cumplir para la **extensión** en una relación.
- Las extensiones que satisfacen las restricciones de DFs se denominan **extensiones legales** o **estados legales** debido a que obedecen las restricciones de la DF.

A	B	C	D
a1	b1	c1	d1
a1	b2	c1	d2
a2	b2	c2	d2
a2	b3	c2	d3
a3	b3	c2	d4

Llaves de las relaciones

Una **llave** puede definirse como un conjunto de atributos $\{A_1, A_2, \dots, A_n\}$ tales que:

- Determinan funcionalmente cualquier otro atributo de la relación. Es decir, es imposible para dos *tuplas* distintas de R coincidir en todos los atributos $\{A_1, A_2, \dots, A_n\}$.
- Ningún subconjunto propio de $\{A_1, A_2, \dots, A_n\}$ determina funcionalmente a los otros atributos de R, es decir, debe ser mínimo.

Una **superllave** es un conjunto de atributos que contiene una **llave**.



Reglas de inferencia de Armstrong

- Fueron desarrolladas por **William W. Armstrong** en 1974: *Dependency Structures of Data Base Relationships*
- Se trata de un conjunto de reglas que permiten deducir todas las dependencias funcionales que tienen lugar en un conjunto de atributos dados, como consecuencia de aquellas que se asumen como ciertas a partir del conocimiento del problema.
- Este resultado permite establecer un conjunto de algoritmos sencillos para:
 1. *Encontrar el conjunto cerrado de un conjunto de dependencias funcionales.*
 2. *Encontrar la equivalencia lógica de esquemas.*
 3. *Deducción de dependencias.*
 4. *Calcular las llaves de un esquema.*

1. **Regla de la reflexividad:** Si $Y \subseteq X$, entonces $X \rightarrow Y$
2. **Regla del aumento:** $\{X \rightarrow Y\} \Rightarrow XZ \rightarrow YZ$
3. **Regla de la transitividad:** $\{X \rightarrow Y, Y \rightarrow Z\} \Rightarrow X \rightarrow Z$
4. **Regla de la descomposición:** $\{X \rightarrow YZ\} \Rightarrow X \rightarrow Y$ y que $X \rightarrow Z$
5. **Regla de la unión:** $\{X \rightarrow Y, X \rightarrow Z\} \Rightarrow X \rightarrow YZ$
6. **Regla de la pseudo-transitividad:** $\{X \rightarrow Y, WY \rightarrow Z\} \Rightarrow WX \rightarrow Z$

Ejemplo

Sean $R = (A, B, C, G, H, I)$ y $F = \{A \rightarrow B, A \rightarrow C, CG \rightarrow H, CG \rightarrow I, B \rightarrow H\}$

Algunos miembros de F^+ serían:

- Unión: **A → BC, CG → HI**
- Pseudo-transitividad: **AG → I, AG → H**
- Aumento: **AG → CG**

Cerradura del conjunto de atributos

Para crear las dependencias funcionales de un esquema, requerimos:

- **Especificar a partir de la semántica de los atributos de R, el conjunto F de DFs.**
- **Usando las reglas de inferencia de Armstrong obtener otras DFs.**

Para obtener todas las dependencias funcionales de manera sistemática:

- **Determinar el conjunto de atributos X del lado izquierdo de alguna DF en F.**
- **Determinar el conjunto X^+ de todos los atributos que son dependientes de X.**

Dado $\{A_1, A_2, \dots, A_n\}$ un conjunto de atributos y F un conjunto de DFs, la **cerradura del conjunto de atributos** ($\{A_1, A_2, \dots, A_n\}^+$) bajo las dependencias en F es el conjunto de atributos B tales que cada relación que satisface todas las dependencias en F también satisface $A_1, A_2, \dots, A_n \rightarrow B$

Algoritmo para calcular X+ bajo F

Repetir

anteriorX+ = X+;

Para cada Y → Z en F hacer

Sí Y ⊆ X+ entonces X+ = X+ ∪ Z;

hasta que anteriorX+ = X+;

Ejemplos:

1. Sean R = {A,B,C,G,H,I} y F = {A → B,A → C,CG → H,CG → I,B → H}, se tiene que:

$$\{AG\} += \{ABCDEFGI\}$$

1. Sea R = {A,B,C,D,E,F} y F = {AB → C,BC → D,D → E,CF → B}, se tiene que {AB} += {ABCDE}, lo que implica que:

$$AB \rightarrow ABCDE$$

Dependencias deducidas

Si queremos probar si una dependencia funcional $A_1, A_2, \dots, A_n \rightarrow B$, se deduce de un conjunto de dependencias F, debemos calcular $\{A_1, A_2, \dots, A_n\}^+$, si B está ahí, entonces la DF si es deducida del conjunto F, en caso contrario no es deducida de F.

Por ejemplo, sea $R = \{A, B, C, D, E, F\}$ y $F = \{AB \rightarrow C, BC \rightarrow D, D \rightarrow E, CF \rightarrow B\}$

- Probar que $AB \rightarrow D$

Se empieza por calcular $\{A, B\}^+ = \{A, B, C, D, E\}$, como $D \in \{A, B\}^+$ entonces ésta si es deducida.

- Probar que $D \rightarrow A$

Se tiene que $D^+ = \{D, E\}$, como A no está en D^+ , entonces la DF no se deduce de F.

Y ahora...
¿Cómo normalizo una
Base de Datos?



Una forma de acabar con anomalías como la redundancia es la **descomposición de relaciones**.

Dada una relación $R(A_1, A_2, \dots, A_n)$, se puede descomponer R en dos relaciones $S(B_1, B_2, \dots, B_i)$ y $T(C_1, C_2, \dots, C_j)$, tales que:

1. $\{A_1, A_2, \dots, A_n\} = \{B_1, B_2, \dots, B_i\} \cup \{C_1, C_2, \dots, C_j\}$
2. Las *tuplas* en la relación S son la proyección sobre $\{B_1, B_2, \dots, B_i\}$ de las *tuplas* en R .
3. De manera similar, las *tuplas* en T son la proyección de $\{C_1, C_2, \dots, C_j\}$ sobre R .

Veamos un ejemplo:

nombre_suc	delegación	activo	nom_cliente	nPréstamo	importe
Centro	Cuauhtémoc	1,800 M	Santos	P – 17	200,000
Copilco	Coyoacán	420 M	Gómez	P – 23	400,000
Viveros	Coyoacán	340 M	López	P – 15	300,000
Centro	Cuauhtémoc	1,800 M	Toledo	P – 14	300,000
Eugenia	Benito Juárez	80 M	Santos	P – 93	100,000
Zapata	Benito Juárez	1,600 M	Abril	P – 11	180,000
San Ángel	Álvaro Obregón	60 M	Vázquez	P – 29	240,000
Tlalpan	Tlalpan	740 M	López	P – 16	260,000
Centro	Cuauhtémoc	1,800 M	González	P – 23	400,000
Viveros	Coyoacán	340 M	Rodríguez	P – 25	500,000

Si se descompone la relación en:

1. Una relación **Sucursal** (*nombre_suc*, *delegación*, *activo*, *nom_cliente*) y
2. Una relación **Préstamo** (*nom_cliente*, *nPréstamo*, *importe*)

nombre_suc	Delegación	activo	nom_cliente
Centro	Cuauhtémoc	1,800 M	Santos
Copilco	Coyoacán	420 M	Gómez
Viveros	Coyoacán	340 M	López
Centro	Cuauhtémoc	1,800 M	Toledo
Eugenia	Benito Juárez	80 M	Santos
Zapata	Benito Juárez	1,600 M	Abril
San Ángel	Álvaro Obregón	60 M	Vázquez
Tlalpan	Tlalpan	740 M	López
Centro	Cuauhtémoc	1,800 M	González
Viveros	Coyoacán	340 M	Rodríguez

nom_cliente	nPréstamo	importe
Santos	P – 17	200,000
Gómez	P – 23	400,000
López	P – 15	300,000
Toledo	P – 14	300,000
Santos	P – 93	100,000
Abril	P – 11	180,000
Vázquez	P – 29	240,000
López	P – 16	260,000
González	P – 23	400,000
Rodríguez	P – 25	500,000

1. Encontrar todas las sucursales que tienen préstamos con importe menor a \$200,000:
 - Con el esquema original:

nombre_suc	delegación	activo	nom_cliente	nPréstamo	importe
Centro	Cuauhtémoc	1,800 M	Santos	P - 17	200,000
Copilco	Coyoacán	420 M	Gómez	P - 23	400,000
Viveros	Coyoacán	340 M	López	P - 15	300,000
Centro	Cuauhtémoc	1,800 M	Toledo	P - 14	300,000
Eugenia	Benito Juárez	80 M	Santos	P - 93	100,000
Zapata	Benito Juárez	1,600 M	Abril	P - 11	180,000
San Ángel	Álvaro Obregón	60 M	Vázquez	P - 29	240,000
Tlalpan	Tlalpan	740 M	López	P - 16	260,000
Centro	Cuauhtémoc	1,800 M	González	P - 23	400,000
Viveros	Coyoacán	340 M	Rodríguez	P - 25	500,000

Eugenia y Zapata

- Con el esquema fraccionado:

nombre_suc	Delegación	activo	nom_cliente
Centro	Cuauhtémoc	1,800 M	Santos
Copilco	Coyoacán	420 M	Gómez
Viveros	Coyoacán	340 M	López
Centro	Cuauhtémoc	1,800 M	Toledo
Eugenia	Benito Juárez	80 M	Santos
Zapata	Benito Juárez	1,600 M	Abril
San Ángel	Álvaro Obregón	60 M	Vázquez
Tlalpan	Tlalpan	740 M	López
Centro	Cuauhtémoc	1,800 M	González
Viveros	Coyoacán	340 M	Rodríguez

Eugenia, Zapata y Centro

nom_cliente	nPréstamo	importe
Santos	P – 17	200,000
Gómez	P – 23	400,000
López	P – 15	300,000
Toledo	P – 14	300,000
Santos	P – 93	100,000
Abril	P – 11	180,000
Vázquez	P – 29	240,000
López	P – 16	260,000
González	P – 23	400,000
Rodríguez	P – 25	500,000

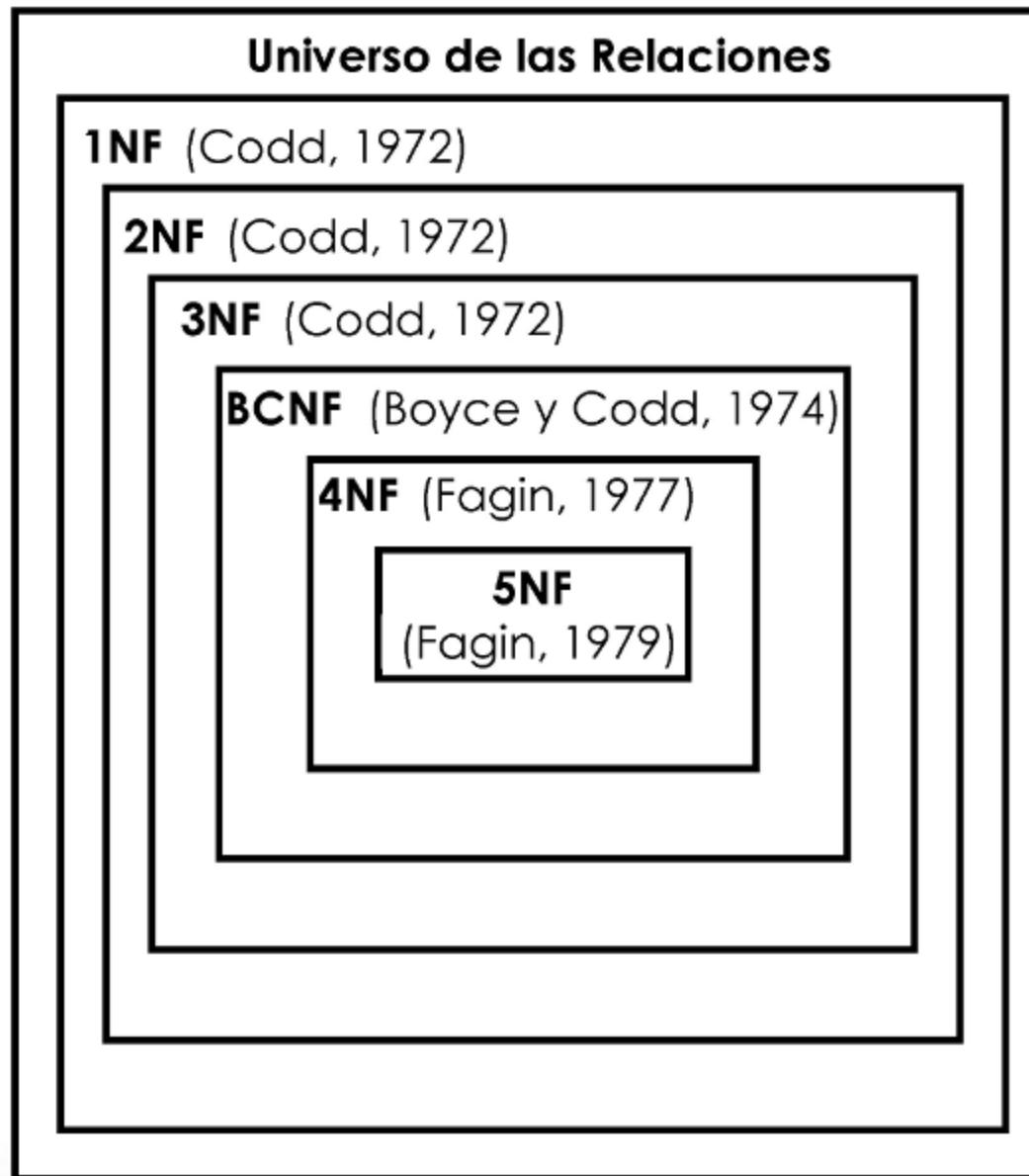
2. Indicar los préstamos que se tienen en cada sucursal el cliente **Santos**:

$$\sigma_{\text{nom_cliente} = \text{'Santos'}} (\textbf{Sucursal} \bowtie \textbf{Prestamo})$$

nom_suc	delegación	nom_cliente	nPréstamo	importe
Centro	Cuauhtémoc	Santos	P – 17	200,000
Centro	Cuauhtémoc	Santos	P – 93	100,000
Eugenia	Coyoacán	Santos	P – 17	200,000
Eugenia	Coyoacán	Santos	P – 93	100,000

Ambas relaciones tienen a *nom_cliente* en común, así que para juntarlas se usa este atributo que no es adecuado puesto que cliente puede tener varios préstamos no necesariamente en la misma sucursal.

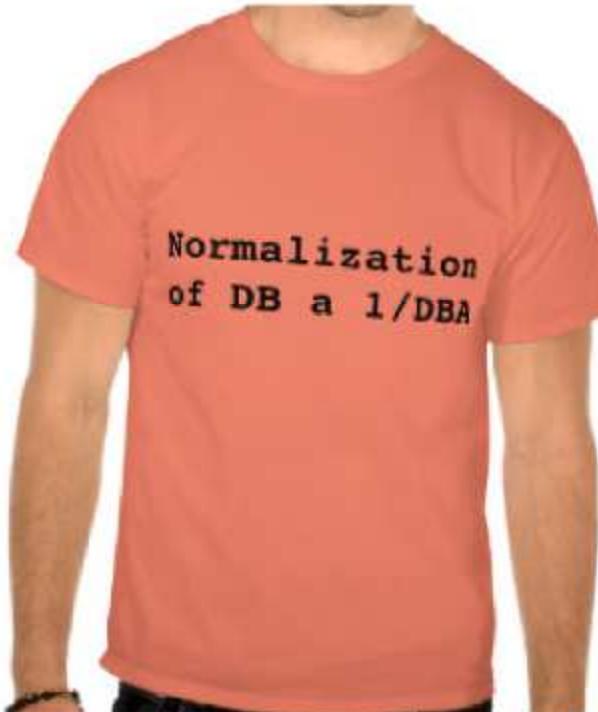
- La **Normalización** es una técnica desarrollada inicialmente por **E.F. Codd en 1972**, para diseñar la estructura lógica de una BD en el modelo relacional.
- Se trata de un proceso en el cual se van comprobando el cumplimiento de una serie de reglas (restricciones) por parte de un esquema de relación.
- Cada regla que se cumple, aumenta el grado de normalización del esquema.
- Cuando una regla no cumple, el esquema de relación se debe descomponer en varios esquemas que sí la cumplan por separado.



Objetivos de la normalización

Para normalizar una BD se desea:

- Eliminar la duplicidad de información.
- Que las relaciones fraccionadas tengan un **join** sin pérdida.
- Conservar las dependencias funcionales.



Forma Normal de Boyce-Codd

Una relación R está en BCNF si y sólo si en toda DF no trivial $A_1, A_2, \dots, A_n \rightarrow B$ para R , se tiene que $\{A_1, A_2, \dots, A_n\}$ es superllave de R .

Por ejemplo:

- La relación $C(\text{nombre}, \text{calle}, \text{ciudad})$ con

$$\text{DF} = \{\text{nombre} \rightarrow \text{calle ciudad}\}$$

- La relación $S(\text{nombre}, \text{no_préstamo})$ con

$$\text{DF} = \{\text{nombre} \rightarrow \text{no_préstamo}\}$$

...Forma Normal de Boyce-Codd

Cualquier relación de dos atributos A y B está en BCNF:

1. Si no hay DF *no triviales* entonces se mantiene la condición BCNF, debido a que sólo una DF no trivial puede violar esta condición (notar que {A,B} es la única llave).
2. Si se tiene $A \rightarrow B$, pero no $B \rightarrow A$, entonces A es la única llave y cada dependencia no trivial contiene A en la izquierda, por tanto no hay violación a la condición BCNF.
3. Si $B \rightarrow A$ y no se tiene $A \rightarrow B$ es un caso simétrico al anterior.
4. Si se tiene $A \rightarrow B$ y $B \rightarrow A$. Entonces tanto A como B son llaves, y cualquier dependencia tiene al menos uno de ellos en su lado izquierdo, por tanto no puede haber violación de la norma BCNF.

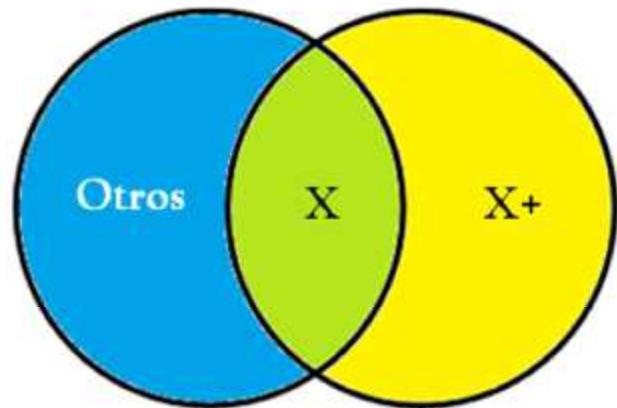
...Forma Normal de Boyce-Codd

Es posible dividir cualquier relación en otras con las siguientes propiedades:

- **Son esquemas de relaciones en BCNF.**
- **Los datos en la relación original se representan fielmente por las relaciones que son resultado de la descomposición.**

La estrategia a seguir es:

1. **Buscar una DF no trivial $X \rightarrow B$ que viole BCNF.**
2. **Calcular X^+ .**
3. **Fraccionar R en $R_1(X^+) \cup R_2((R-X^+) \cup X)$.**
4. **Encontrar las DF para las nuevas relaciones.**



Se debe aplicar la regla de descomposición tantas veces como sea necesario hasta que todas las relaciones estén en BCNF.

Ejemplo: BCNF

La relación $P(\text{nombreSuc}, \text{nombre_cliente}, \text{no_préstamo}, \text{importe})$ con $\text{DF} = \{\text{no_prestamo} \rightarrow \text{importe nombreSuc}\}$ se descompone en:

- **P1(no_prestamo, importe, nombreSuc)**
- **P2(nombre_cliente, no_préstamo)**

Sea R una relación descompuesta en S y otra más. Sea F el conjunto de DFs para R. Para calcular las DF mantenidas en S, hacer:

1. Para cada conjunto X de atributos contenidos en S calcular X^+ .
2. Para cada atributo B tal que:
 - B es un atributo de S,
 - B no está en X, y
 - B está en X^+

se tiene la DF $X \rightarrow B$ en S



**Necesitas normalizar con
BCNF en el examen...**

**Sería una lástima que
ninguna DF violara la forma
normal...**

Recuperación de información

Interesa que la descomposición preserve la información contenida en la relación original.

Consideremos $R(A,B,C)$ con $B \rightarrow C$ que suponemos es una violación a la BCNF,

1. Al descomponer R obtenemos $S(B,C)$ y $T = (A,B)$.
2. Sea $t = (a,b,c)$ una tupla de R .
3. Al proyectarla en la descomposición se obtienen (a,b) para T y (b,c) para S .
4. Al hacer un *join* sobre el atributo común, en este caso B , obtenemos nuevamente t .

Sin embargo, regresar a las *tuplas* iniciales no es suficiente para asegurar que la relación original está realmente representada por la descomposición.

Problemas con la recuperación

Si se tiene la relación con la siguiente extensión:

R =	A	B	C
a	b	c	
d	b	e	

ésta se descompone en las 2 siguientes con su respectiva proyección:

S =	A	B
a	b	
d	b	

T =	B	C
b	c	
b	e	

S \bowtie T =	A	B	C
a	b	c	
a	b	e	
d	b	c	
d	b	e	

¿Son correctas las proyecciones?

La justificación de la no pérdida ni ganancia de información es debido a que se están considerando a las DFs.

Tercera Forma Normal

En ocasiones se puede encontrar que un esquema de relación y sus DF no están en BCNF pero no se desea descomponer más, por ejemplo:

Reservaciones (película, cine, ciudad) con
DF = {cine → ciudad, película ciudad → cine}

Ningún atributo por sí solo es una llave; por otro lado, la pareja {cine, película} sí son llaves, de manera que **cine → ciudad**, viola la **BCNF**.

Si normalizamos esta relación obtenemos:

	cine	ciudad
S =	Real cinema Linterna mágica	D.F. D.F.
T =	Real cinema Linterna mágica	La vida es bella La vida es bella

...Tercera Forma Normal

Ambas relaciones son permisibles de acuerdo a las DF de cada relación, pero al unirlas obtenemos:

	cine	Ciudad	película
$S \bowtie T =$	Real cinema Linterna mágica	D.F. D.F.	La vida es bella La vida es bella

que viola la DF **película ciudad → cine**.

La solución al problema anterior es relajar la condición para la **BCNF**.

...Tercera Forma Normal

Una relación R está en **Tercer Forma Normal (3NF)** con respecto a F , si para toda dependencia no trivial $A_1, A_2, \dots, A_n \rightarrow B$, se tiene que:

1. **El lado izquierdo (A_1, A_2, \dots, A_n) es una superllave o bien,**
2. **El lado derecho, B , es miembro de alguna llave candidata de R**

El segundo punto es el que permite una dependencia como **cine \rightarrow ciudad** del ejemplo anterior, porque ciudad es miembro de una llave.

Siempre es posible descomponer un esquema de relación sin pérdida de información en esquemas que están en 3NF y permiten que se verifiquen todas las DFs.

Si estas relaciones no están en BCNF, se tendrá un poco de redundancia en el esquema.

Atributos superfluos

A es un **atributo superfluo** si se puede eliminar de la DF sin que se altere la cerradura de F.

Sea $\alpha \rightarrow \beta$ una DF en F y A un atributo, A es superfluo si:

1. Si A esta en α (**superfluo por la izquierda**).
2. Si A esta en β (**superfluo por la derecha**)

Equivalencia de conjuntos de DFs

Dos conjuntos de dependencias funcionales, F_1 y F_2 son equivalentes si:

$$F_1 \vDash F_2 \text{ y } F_2 \vDash F_1$$

Por ejemplo:

Sea $F = \{A \rightarrow B, B \rightarrow C, AC \rightarrow D\}$

- Si $F_1 = \{A \rightarrow B, B \rightarrow C, A \rightarrow D\}$ es equivalente a F , ya que:
 $\{A\}^+ = \{ABCD\}$ y $\{AC\}^+ = \{ABCD\}$
- Si $F_2 = \{A \rightarrow B, B \rightarrow C, C \rightarrow D\}$ no es equivalente a F , ya que:
 $\{C\}^+ = \{CD\}$

Un conjunto F de dependencia funcionales es **mínimo** si

1. No tiene **atributos superfluos**
2. Cada lado izquierdo de las DF de F es único, es decir, no existen $\alpha_1 \rightarrow \beta_1, \alpha_2 \rightarrow \beta_2$ tales que $\alpha_1 = \alpha_2$.

El algoritmo para calcular el conjunto **F'** equivalente a **F** que sea mínimo es:

Repetir

1. Aplicar la regla de la unión a relaciones tales que $\alpha_1 \rightarrow \beta_1, \alpha_1 \rightarrow \beta_2$, para obtener $\alpha_1 \rightarrow \beta_1\beta_2$ y sustituir con esta última las dependencias funcionales con igual lado izquierdo.
2. Eliminar los atributos superfluos de las dependencias funcionales.

Hasta que ya no haya ningún cambio.

Algoritmo de síntesis: 3NF

Su objetivo es descomponer R con dependencias funcionales F , en relaciones que satisfagan la **3NF**.

1. *Hacer F mínimo*
2. *Para toda DF en F mínimo:*
 - a. *Crear una relación que contenga sólo los atributos de las DF.*
 - b. *Eliminar un esquema si es subconjunto de otro.*
3. *Si no existen esquemas que contengan llaves candidatas, crear una relación con esos atributos.*

Sea $R(A, B, C, D)$ y $F = \{A \rightarrow B, B \rightarrow C, AC \rightarrow D\}$. Normalizar con **3NF**:

Vamos a encontrar el **conjunto mínimo** equivalente:

- Podemos observar que no hay lados izquierdos en común, entonces comenzamos por determinar atributos superfluos
- Tomamos la DF $AC \rightarrow D$ y verificamos si **C es superfluo del lado izquierdo**:

$$A \rightarrow D \Rightarrow \{A\}^+ = \{ABCD\} \therefore C \text{ es superfluo}$$

$$\Rightarrow F' = \{A \rightarrow B, B \rightarrow C, A \rightarrow D\}$$

Unimos lados izquierdos en común:

$$F' = \{A \rightarrow BD, B \rightarrow C\}$$

- Verificamos ahora atributos superfluos, pero, ahora del **lado derecho**.

¿B es superfluo? $A \rightarrow BD \Rightarrow A \rightarrow D \Rightarrow F' = \{A \rightarrow D, B \rightarrow C\}$

$\{A\}^+ = \{AD\} \therefore B \text{ no es superfluo}$

¿D es superfluo? $A \rightarrow BD \Rightarrow A \rightarrow B \Rightarrow F' = \{A \rightarrow B, B \rightarrow C\}$

$\{A\}^+ = \{ABC\} \therefore D \text{ no es superfluo}$

Por lo tanto,

$F' = \{A \rightarrow BD, B \rightarrow C\}$ es un **conjunto mínimo**

El siguiente paso consiste en crear una relación que contenga solo los atributos de cada una de las DF:

S(A,B,D) y T(B,C)

De lo anterior podemos observar que no existen esquemas que sean subconjunto de otro, entonces, calculamos la llave:

{A}+ = {ABCD}

Como **A** ya esta contenida en la **S**, ya no es necesario llegar al paso tres del algoritmo y la normalización en **3NF** es:

R1(A,B,D) y **R2(B,C)**

Dependencias Multivaluadas

Sea $R(\text{nombre}, \text{dirección}, \text{teléfono}, \text{afición})$ con:

$\text{nombre} \rightarrow \text{dirección}$

nombre	dir	tel	afición
n1	d1	t1	h1
n1	d1	t1	h2
n1	d1	t1	h3
n1	d1	t2	h1
n1	d1	t2	h2
n1	d1	t2	h3

Como se puede observar, la llave es **nombre teléfono afición**, de manera que una violación a la BCNF es $\text{nombre} \rightarrow \text{dirección}$. Podemos dividir la relación en $S(\text{nombre}, \text{dirección})$ y $T(\text{nombre}, \text{teléfono}, \text{afición})$ las cuales se encuentran en BCNF:

nombre	dir
n1	d1

nombre	tel	afición
n1	t1	h1
n1	t1	h2
n1	t1	h3
n1	t2	h1
n1	t2	h2
n1	t2	h3

...Dependencias Multivaluadas

Existe una **dependencia multivaluada (DMV)** $A_1A_2\dots A_n \rightarrow\!\!> B_1B_2\dots B_m$ si para cada par de tuplas t_1 y t_2 de la relación R , que coinciden en todos los valores de las A 's, podemos encontrar una tupla t_3 tal que coincida con:

1. t_1 y t_2 en las A 's
2. t_1 en las B 's y
3. t_2 en todos los atributos de R que no están ni en A ni en B

	A's	B's	Otros
t_1	Yellow	Green	
t_2	Yellow		Pink
t_3	Yellow	Green	Pink

- Las DF excluyen ciertas tuplas de una relación, si $A \rightarrow B$ entonces no puede haber dos tuplas con el mismo valor de A y diferente de B.
- Las **DMV's** permiten que otras tuplas de esta forma se presenten en la relación. Por ejemplo:

préstamo	nombre_cliente	calleNum	Ciudad
P23	Gómez	Clavel 25-A-205	Cuernavaca
P23	Gómez	Insurgentes 4141	México D.F.
P93	Pérez	Juárez 85	Oaxaca

Si el cliente Gómez obtuviera otro préstamo, digamos el **P27**, se tendrían que agregar dos tuplas:

préstamo	nombre_cliente	calleNum	Ciudad
P23	Gómez	Clavel 25-A-205	Cuernavaca
P23	Gómez	Insurgentes 4141	México D.F.
P27	Gómez	Clavel 25-A-205	Cuernavaca
P27	Gómez	Insurgentes 4141	México D.F.
P93	Pérez	Juárez 85	Oaxaca

Se requiere que esta relación tenga la **DMV**:

nombre → calleNum ciudad

Se debe verificar esta condición para **TODOS** los pares de tuplas que coincidan con el nombre, no solo con un par.

Sean $A = \{A_1, A_2, \dots, A_n\}$ y $B = \{B_1, B_2, \dots, B_n\}$ conjuntos de atributos de una relación R, las **DMVs** satisfacen las siguientes reglas:

- **Complemento:**

Si se tiene $A \twoheadrightarrow B$ para alguna relación, entonces se debe tener $A \twoheadrightarrow C$, donde C son todos los atributos de R que no son ni A s, ni B s.

- **Transitividad:**

Si se tienen las **DMVs** $A \twoheadrightarrow B$, $B \twoheadrightarrow C$, entonces también se tiene que $A \twoheadrightarrow C$.

- **Replicación.** Toda **DF** es una **DMV**.

NOTA:

Las **DMV** no obedecen las reglas de **división/combinación**.

Sea $R = (A, B, C, D, G, H, I)$ y $DMV = \{A \twoheadrightarrow B, B \twoheadrightarrow HI, CG \twoheadrightarrow H\}$

Aplicando las reglas se puede obtener:

- $A \twoheadrightarrow CDGHI$; por la **regla del complemento**.
- $A \twoheadrightarrow HI$; por la **regla de transitividad** aplicada a la primera y segunda dependencias.
- $\{B \twoheadrightarrow H, B \twoheadrightarrow I\} \Rightarrow$ No es posible pues **no hay división**.

Una $DMV A_1A_2\dots A_n \twoheadrightarrow B_1B_2\dots B_n$ es **no trivial** si:

1. Ninguna de las B 's está contenida en las A 's
2. R tiene más atributos que las A 's y las B 's.

- La redundancia que proviene de las DMV, puede eliminarse si utilizamos en un nuevo algoritmo de descomposición para las relaciones.
- La **Cuarta Forma Normal** elimina todas las DMV no triviales, lo mismo que las DF que violan la **BCNF**. En consecuencia, las relaciones descompuestas resultantes no tienen ni la redundancia de las DF ni de las DMV.
- Criterio para **4NF**:
*Una relación **R** está en **4FN** si toda DMV no trivial $A_1, A_2 \dots A_n \Rightarrow B_1, B_2 \dots B_m$ tiene que $\{A_1, A_2 \dots A_n\}$ es una superllave*
- Es importante hacer notar que las nociones de **llave** y **superllave** dependen sólo de las DFs. La **4NF** es una generalización de la **BCNF** debido a que toda **DF** es una **DMV**.
- Por lo tanto, toda violación a la **BCNF** es una violación a la **4NF**, pero al revés no es cierto.

Objetivo: Eliminar la redundancia debida al efecto multiplicativo de las DMVs.

1. Encontrar una violación a la **4NF**, es decir, $A_1A_2\dots A_n \twoheadrightarrow B_1B_2\dots B_m$ donde el conjunto de las **A's** no forma una **superllave**.
2. Dividir la relación en dos esquemas:
 - El que contiene las **A's** y las **B's**
 - El que contiene las **A's** y los atributos de **R** que no están entre las **B's**.
3. Si alguno de los esquemas no estuviera en **4NF**, regresar al **paso 1**.

En este caso, no hay pruebas análogas a las de la cerradura de atributos para DFs.

Sea **BC(préstamo, nombre, calleNum, ciudad)** con las dependencias:

$$\{\text{nombre} \rightarrow \text{préstamo}, \text{nombre} \twoheadrightarrow \text{calleNum } \text{ciudad}\}$$

1. **Buscamos una DMV que viole la 4NF** (lado izquierdo no es una superllave. Para determinar la llave, solo consideramos las DF)

Para este caso, la llave es **nombre calleNum ciudad** (esto es porque **nombre** ya determina a **préstamo** y solo falta considerar **calleNum** y **ciudad**)

Por lo tanto, la DMV que viola es: **nombre \twoheadrightarrow calleNum ciudad**

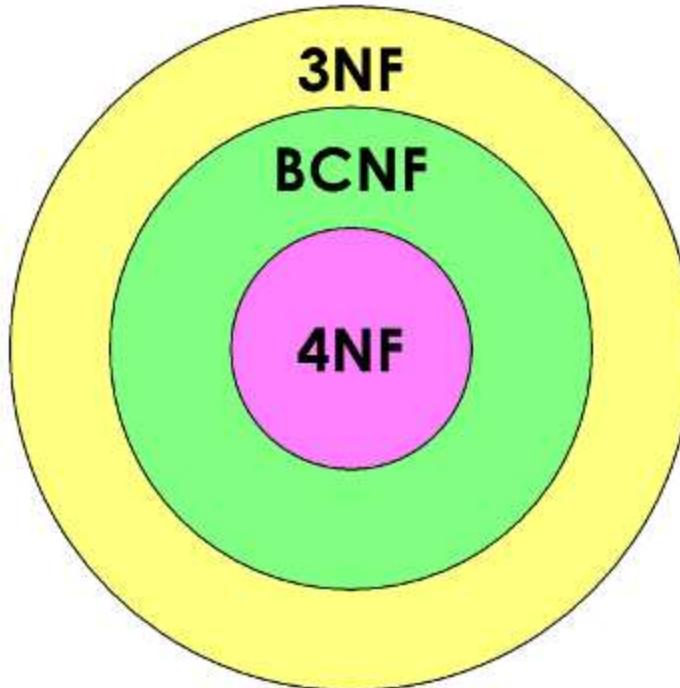
- Dividimos la relación en:

S(nombre, calleNum, ciudad) se cumple **nombre → calleNum ciudad**

T(nombre, préstamo) se cumple **nombre → préstamo**

- Como se puede observar, en **S**, la DMV **nombre → calleNum ciudad** es trivial y además no hay más DMV que violen la **4FN**, por lo tanto, esta relación ya está normalizada.
- En el caso de **T**, la dependencia **nombre → préstamo** es trivial y no hay **DMV** que violen la **4FN**.

Relación entre Formas Normales



Propiedad	3NF	BCNF	4NF
Elimina la redundancia por dependencias funcionales	La mayor parte	Sí	Sí
Elimina la redundancia debida a dependencias multivaluadas	No	No	Sí
Conserva las dependencias funcionales	Sí	Quizá	Quizá
Conserva las dependencias multivaluadas	Quizá	Quizá	Quizá