



Teoría formal de la normalización de esquemas relacionales.

Definición formal de las tres primeras Formas Normales

Normalización de esquemas relacionales

Motivación

Sea la **BD** de proveedores y partes, con 3 relaciones: **S**, **P** y **SP**

S

S#	SNOMBRE	SITUACIÓN	CIUDAD
S1	Salazar	20	Londres
S2	Jaimes	10	París
S3	Bernal	30	París
S4	Corona	20	Londres
S5	Aldana	30	Atenas



P

P#	PNOMBRE	COLOR	PESO	CIUDAD
P1	Tuerca	Rojo	12	Londres
P2	Perno	Verde	17	París
P3	Birlo	Azul	17	Roma
P4	Birlo	Rojo	14	Londres
PS	Leva	Azul	12	París
P6	Engranaje	Rojo	19	Londres



SP



S#	P#	CANT
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P3	200
S4	P2	200
S4	P4	300
S4	P5	400

Recordemos: Modelo relacional

Tabla → **Relación**

S

Definición de la tabla → **Esquema de relación**

S (S#, SNOMBRE, SITUACION, CIUDAD)

Columna de la tabla → **Atributo**

CIUDAD

Filas de la tabla → **Tupla**

(S1, Salazar, 20, Londres)



Objetivo del diseño

- ♦ generar un **conjunto de esquemas de relaciones** que permitan almacenar información **sin redundancia** innecesaria,
- ♦ pero que a la vez permita **recuperar información fácilmente**.

Un enfoque es **diseñar esquemas**
que tengan una **forma normal** apropiada.

Se definirán las formas normales usando el **concepto de DF**.

Representación de información

Problemas y soluciones

Los **defectos** que puede tener una BD mal diseñada son:

- repetición de información
- incapacidad para representar cierta información
- pérdida de información

Solución:

descomponer el esquema de relación con problemas en
varios esquemas de relaciones.

S, P y SP → Parece un **diseño correcto**

Pero... si se coloca **CIUDAD** del proveedor **en SP**
(y no en S),

se obtiene SP'

S#	CIUDAD	P#	CANT
S1	Londres	P1	300
S1	Londres	P2	200
S1	Londres	P3	400
S1	Londres	P4	200
S1	Londres	PS	100
S1	Londres	P6	100
S2	París	P1	300
S2	París	P2	400

S, P y SP' → Es un mal diseño



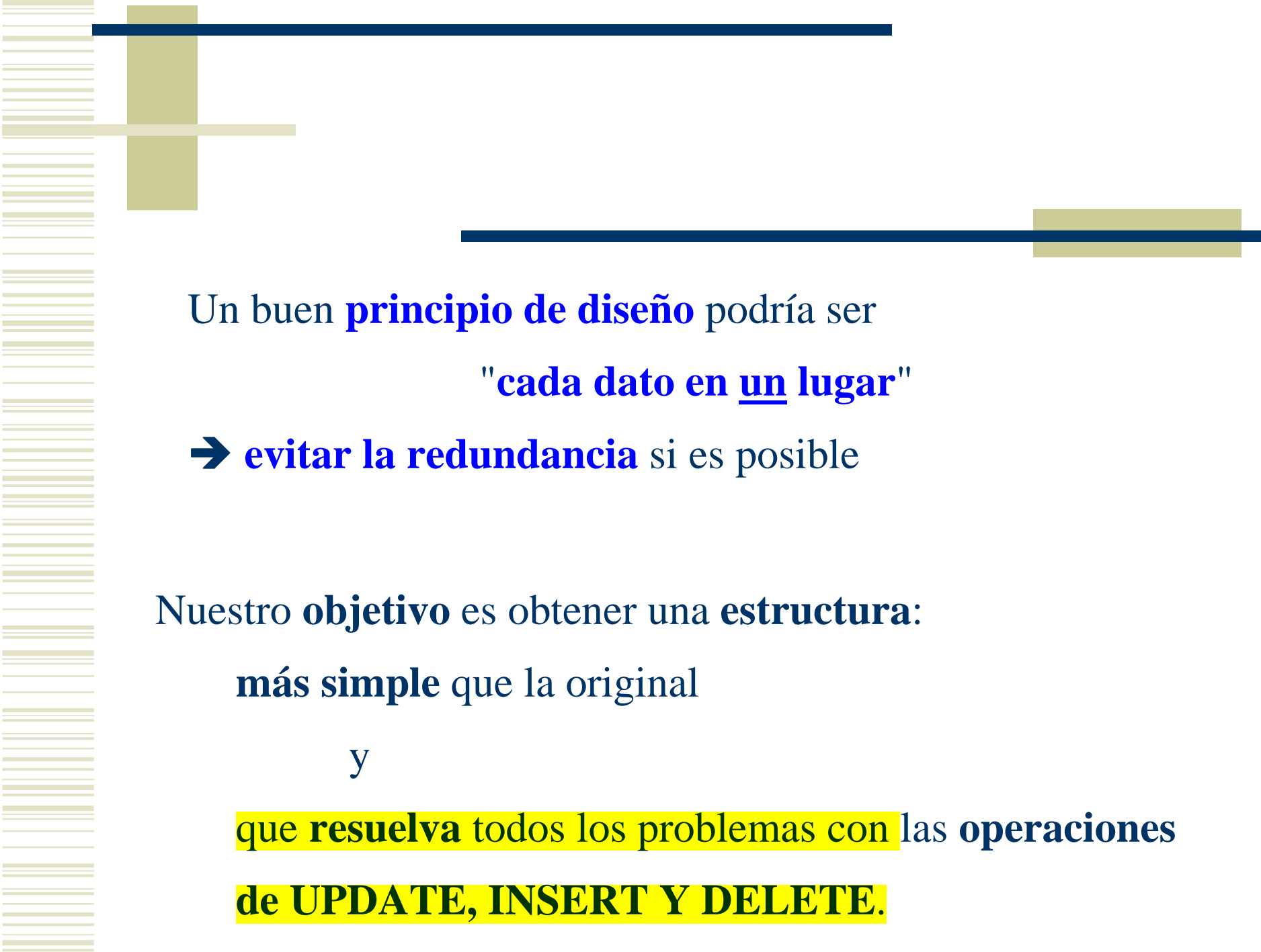
Problemas de la relación SP'

La BD contiene un **alto grado de redundancia**

- ♦ La **CIUDAD** de un proveedor aparece **tantas veces** como **envíos** haya de ese proveedor.

Esta redundancia provoca **problemas**:

- ♦ Después de una **actualización**, podría quedar en una tupla que **S1** está situado en **Londres** y en otra tupla en **Amsterdam**.



Un buen **principio de diseño** podría ser
"cada dato en un lugar"

→ **evitar la redundancia** si es posible

Nuestro **objetivo** es obtener una **estructura**:
más simple que la original

y

que **resuelva** todos los problemas con las **operaciones**
de UPDATE, INSERT Y DELETE.

Formas Normales

Una relación está en una cierta forma normal si **satisface un cierto conjunto de restricciones**.

- ♦ Se ha definido un **gran número de formas normales**.

Codd definió la primera, segunda y tercera formas normales

(1NF, 2NF, 3NF)

El **diseñador** de una BD debe tratar de lograr un **diseño con relaciones por lo menos en 3NF**.

Dependencia Funcional (DF)

Dada una relación R, el atributo **Y** de R **depende funcionalmente** del atributo **X** de R

$$X \rightarrow Y$$

si y sólo si **un solo valor Y** en R está **asociado a cada valor X** en R en cualquier momento.

Si dos tuplas **coinciden** en el valor de **X**
deben **coincidir** en el valor de **Y**

Ej. Código Postal \rightarrow Ciudad

Dependencia Funcional (DF)

En nuestra BD

$S\# \rightarrow SNOMBRE$

$S\# \rightarrow SITUACION$

$S\# \rightarrow CIUDAD$

Porque **dado un valor** de $S\#$, existe sólo un valor de $SNOMBRE$,
de $SITUACIÓN$ y de $CIUDAD$.

COLOR no determina PESO

Porque **para cada color** no hay un solo peso

- $P1$ es **roja** y tiene un peso de **12**,
- $P6$ también es **roja** pero tiene un peso de **19**.

Si X es clave candidata de $R \rightarrow$ todos los atributos de R deben por fuerza depender funcionalmente de X .

Dependencia funcional completa

$X \rightarrow Y$ es una **DF completa** si:

- Y depende funcionalmente de X
- Y no depende de ningún subconjunto propio de X.

Ejemplo:

En SP' $(S\#,P\#) \rightarrow CIUDAD$

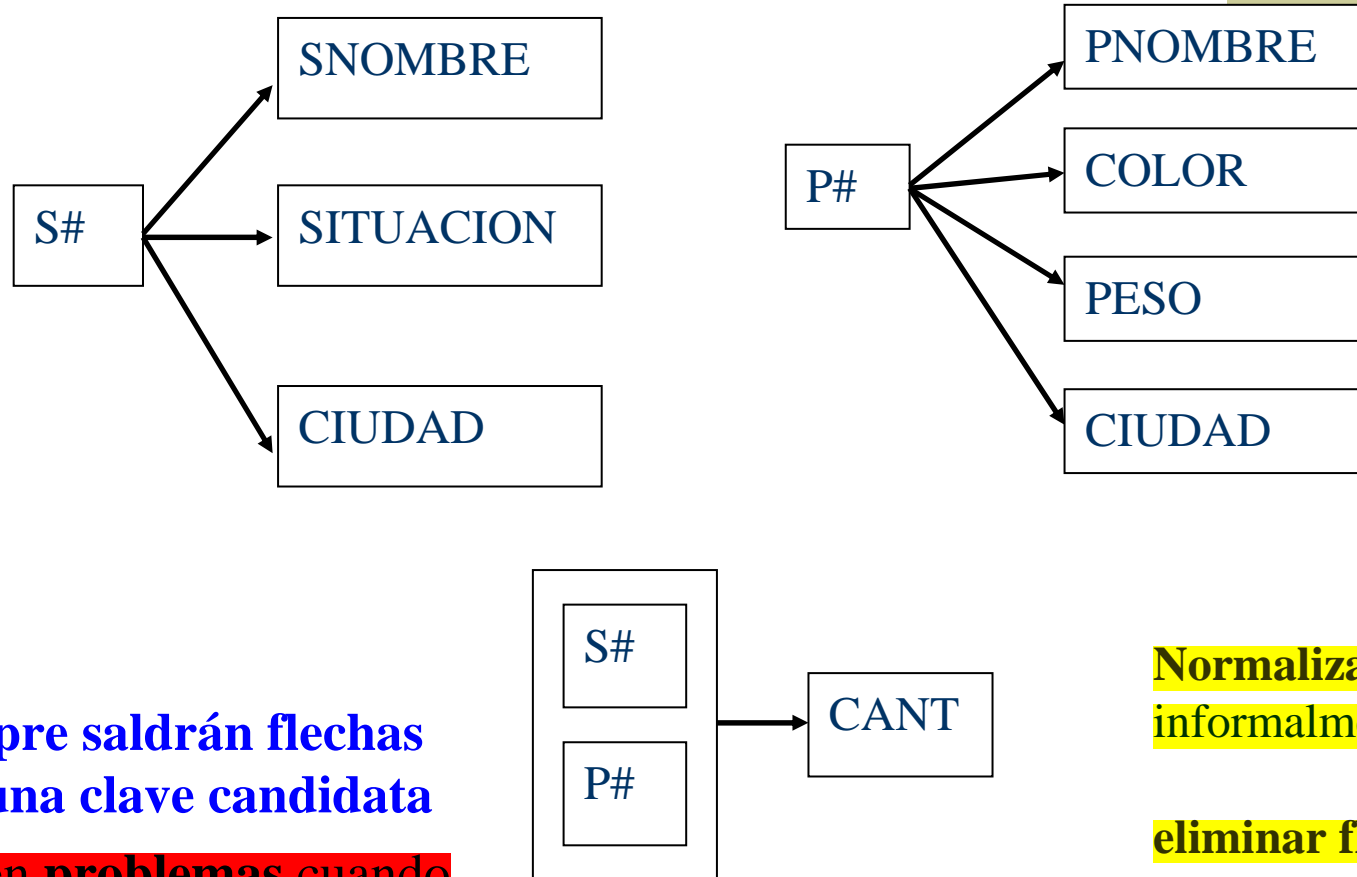
Pero **no** es una DF completa, porque

$S\# \rightarrow CIUDAD$

SP'

S#	CIUDAD	P#	CANT
S1	Londres	P1	300
S1	Londres	P2	200
S1	Londres	P3	400
S1	Londres	P4	200
S1	Londres	PS	100
S1	Londres	P6	100
S2	París	P1	300
S2	París	P2	400

Diagrama de DF



- Siempre saldrán flechas de una clave candidata
- Surgen problemas cuando existen otras flechas.

Normalizar, informalmente, es

eliminar flechas que no salgan de claves candidatas.

Solución de anomalías mediante las formas normales

Primera Forma Normal(1NF)

Definición informal:

Una relación está en 1NF si en cada intersección de fila y columna de una tabla siempre **existe un solo valor** , nunca una lista.

Definición formal:

Una relación está en 1NF si y sólo si **todos los dominios simples subyacentes contienen sólo valores atómicos**.

Una relación en 1NF presenta varios problemas

Supongamos tener en vez
de S y SP una sola
relación **PRIMERA**

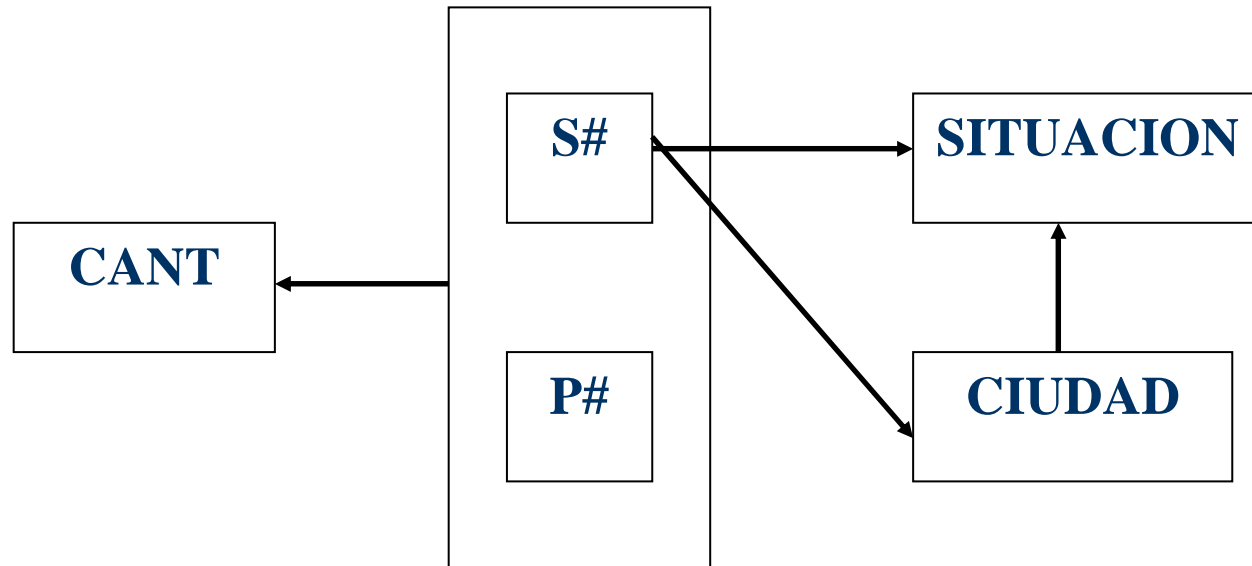
Agregamos una
restricción adicional:

CIUDAD → SITUACION

clave primaria : (S#,P#).

S#	SITUACIÓN	CIUDAD	P#	CANT
S1	20	Londres	P1	300
S1	20	Londres	P2	200
S1	20	Londres	P3	400
S1	20	Londres	P4	200
S1	20	Londres	P5	100
S1	20	Londres	P6	100
S2	10	París	P1	300
S2	10	París	P2	400
S3	10	París	P2	200
S4	20	Londres	P2	200
S4	20	Londres	P4	300
S4	20	Londres	P5	400

El diagrama de DF es



- hay flechas que salen de la PK junto con ciertas **flechas adicionales**
- esas **flechas adicionales causan todos los problemas.**



Anomalías de Actualización

Las **redundancias** son obvias.

Las **redundancias** provocan "**anomalías de actualización**".

- Problemas con las tres operaciones de actualización:

INSERT, DELETE y UPDATE.

Anomalías de Actualización

INSERT:

No podemos insertar el hecho de que un **proveedor está situado en una ciudad** si ese proveedor no suministra por lo menos una parte.

PRIMERA no indica que **S5** está situado en Atenas.

Una forma de representarlos sería utilizando **nulos** en **P#**, pero por la **regla de integridad de entidades**:

ningún componente de la clave primaria puede ser nulo.

Anomalías de Actualización

UPDATE:

La ciudad de un proveedor aparece varias veces en PRIMERA

Se podría producir un **resultado inconsistente**.

DELETE:

Si eliminamos la única tupla de PRIMERA de un proveedor, **perdemos** la información de la **ciudad** en la que está situado.

si eliminamos la tupla donde $S\# = S3$ y $P\# = P2$
perdemos la información: **S3 está situado en París.**

Anomalías de Actualización

Problema: PRIMERA contiene **demasiada información**

→ cuando se elimina una tupla, se **elimina demasiado**.

PRIMERA contiene información de **envíos** y **proveedores**

→ **eliminar** un envío hace que se elimine **también** información de **proveedores**.

Solución: “**desempacar**”

- colocar información de envíos en una relación

- colocar información de proveedores en otra

Es decir: colocar **información separada en relaciones separadas**.

Solución a estos tres problemas de actualización

sustituir PRIMERA por

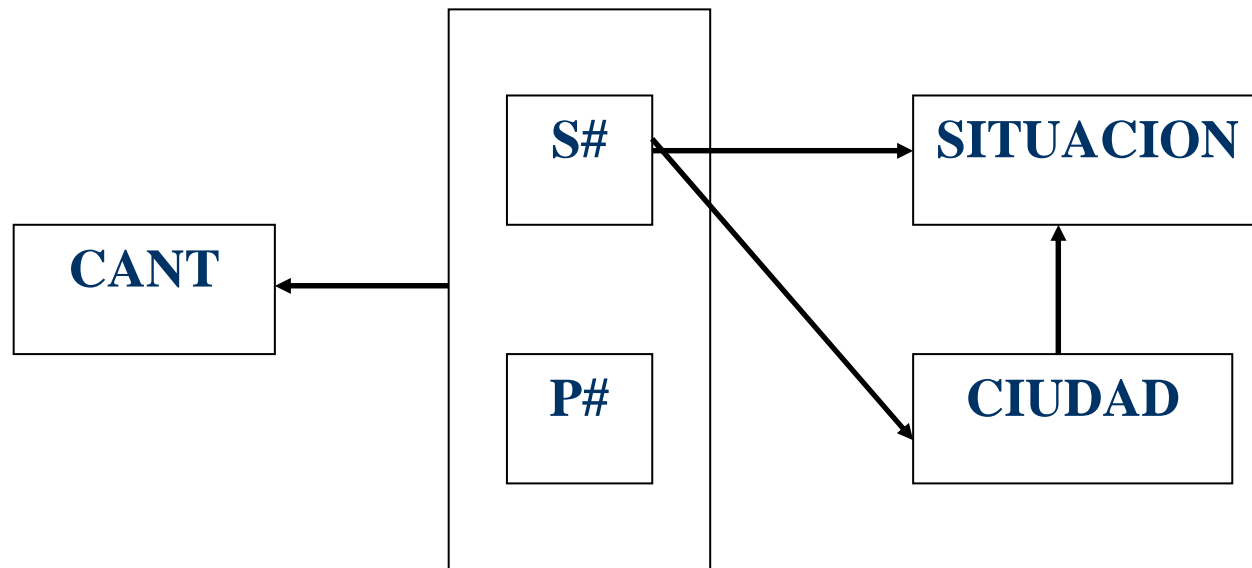
SEGUNDA (S#, SITUACIÓN, CIUDAD)

SP (S#, P#, CANT)

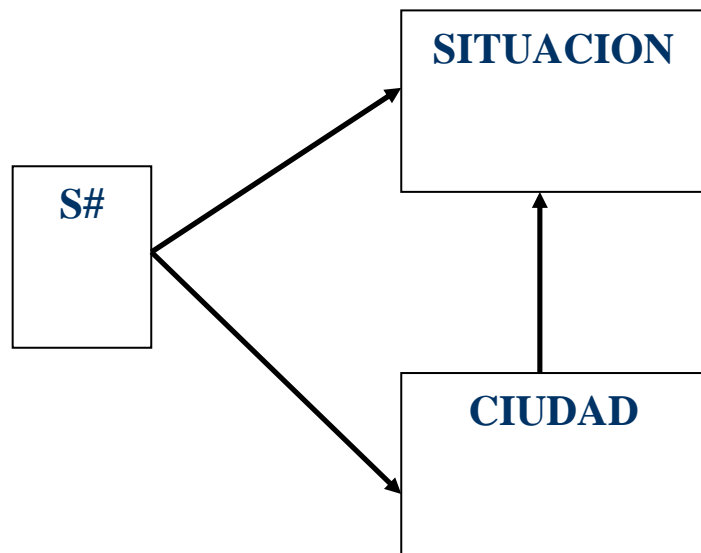
S#	SITUACIÓN	CIUDAD
S1	20	Londres
S2	10	París
S3	10	París
S4	20	Londres
S5	30	Atenas

S#	P#	CANT
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

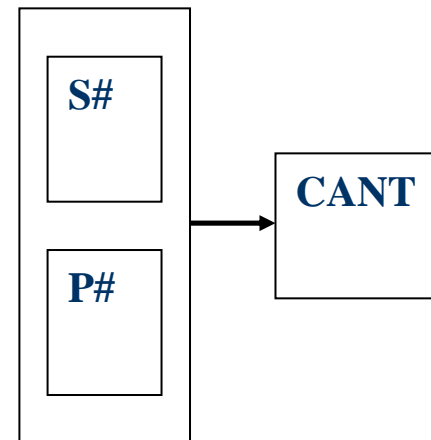
El diagrama de DF era



Los **diagramas DF** de estas relaciones son



SEGUNDA



SP

Esta estructura resuelve los problemas

INSERT:

Podemos insertar la información que

S5 está en Atenas,

aún cuando S5 no suministre una parte.

S#	SITUACIÓN	CIUDAD
S1	20	Londres
S2	10	París
S3	10	París
S4	20	Londres
S5	30	Atenas

S#	P#	CANT
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

Esta estructura resuelve los problemas

DELETE:

Podemos eliminar el envío que conecta a S3 con P2 en SP, sin perder que S3 está en París.

S#	SITUACIÓN	CIUDAD
S1	20	Londres
S2	10	París
S3	10	París
S4	20	Londres
S5	30	Atenas

S#	P#	CANT
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

Esta estructura resuelve los problemas

UPDATE:

La ciudad de un proveedor
aparece una sola vez, no muchas.

S#	SITUACIÓN	CIUDAD
S1	20	Londres
S2	10	París
S3	10	París
S4	20	Londres
S5	30	Atenas

S#	P#	CANT
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400



Esta estructura resuelve los problemas

**Se han eliminado las dependencias no completas,
y con esa eliminación se han resuelto los problemas.**

Segunda Forma Normal

Atributo **primo** es un atributo que forma parte de la clave primaria.

Definición informal:

Una relación está en **2NF** si y sólo si está en **1NF** y todos los **atributos no clave dependen por completo de la clave primaria**

Definición formal:

Una relación R está en **2NF** si está en **1NF** y cada uno de sus **atributos no primos es dependiente funcional completo de cada clave candidata de R**

Segunda Forma Normal

- ♦ SEGUNDA y SP están en 2NF
- ♦ Las claves primarias son $S\#$ y $(S\#,P\#)$
- ♦ Una relación en 1NF pero no en 2NF siempre podrá reducirse a un conjunto equivalente de relaciones en 2NF

Mediante el **procedimiento de normalización**,
una relación en **cierta forma normal** (1NF), se puede **convertir** en un
conjunto de relaciones en una **forma más deseable** (2NF).

Segunda Forma Normal

El procedimiento es **reversible**:

- ♦ **siempre** es posible **tomar la salida** del procedimiento (conj de relaciones 2NF)
- ♦ **y convertirlas otra vez en la entrada** (la relación 1NF).

El proceso de reducción es un proceso de sacar **proyecciones**.

- ♦ El operador de **descomposición** es la **proyección**.
- ♦ El operador de **recomposición** es la **reunión natural**.

→ **no se pierde información** durante el proceso de normalización.

La estructura SEGUNDA y SP todavía causa problemas

S# → SITUACION se obtiene por transitividad (a través de CIUDAD)

- **Las DF transitivas producen anomalías de actualización.**

S#	SITUACIÓN	CIUDAD
S1	20	Londres
S2	10	París
S3	10	París
S4	20	Londres
S5	30	Atenas

La estructura SEGUNDA y SP todavía causa problemas

INSERT: No podemos insertar que una CIUDAD tiene una SITUACIÓN si no hay algún proveedor situado en esa ciudad.

No se puede representar: "Roma tiene una situación de 50"

DELETE: Si eliminamos la única tupla en SEGUNDA de una ciudad, perdemos la información:

de que esa CIUDAD tiene esa SITUACIÓN particular.

Eliminar S5 en SEGUNDA → perder que: situación de Atenas es 30.

UPDATE: La SITUACIÓN de una ciudad aparece en SEGUNDA muchas veces.

Se podrían producir inconsistencias:

cambiar situación de Londres a 20 en una tupla y 30 en otra.

Solución

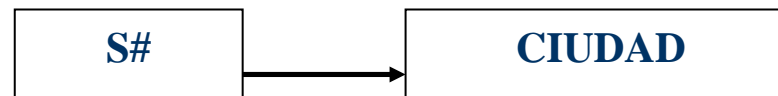
sustituir SEGUNDA por dos proyecciones:

SC (S#, CIUDAD)

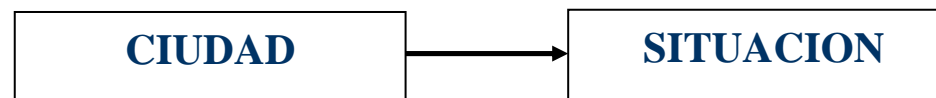
CS (CIUDAD, SITUACION)

Los **diagramas DF** son:

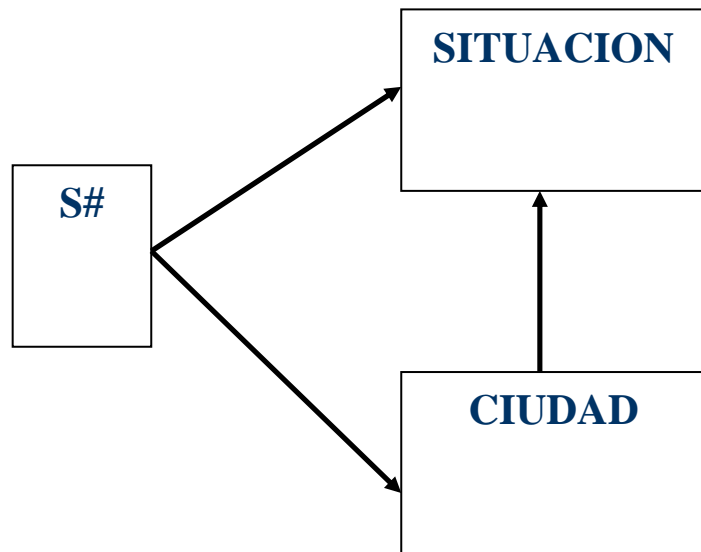
SC



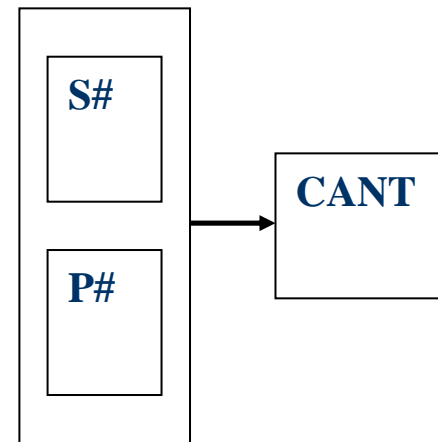
CS



El diagrama de DF era



SEGUNDA



SP

Y las tablas son

CIUDAD	SITUACIÓN
Atenas	30
Londres	20
París	10
Roma	50

S#	CIUDAD
S1	Londres
S2	París
S3	París
S4	Londres
S5	Atenas

Tercera Forma Normal

Definición informal:

Una relación está en 3NF \Leftrightarrow los atributos no clave (si los hay) son

- (a) mutuamente independientes, y
- (b) dependientes por completo de la clave primaria.

Dos atributos son mutuamente independientes si ninguno de ellos depende funcionalmente de cualquier combinación de los otros.

Es decir: esos atributos se pueden actualizar sin tomar en cuenta a los demás.

Definición formal:

Una relación está en 3NF \Leftrightarrow

está en 2NF y

todos los atributos no primos dependen de manera no transitiva de la clave primaria

Tercera Forma Normal

- SC y CS están en 3NF
- **SEGUNDA no** está en 3NF
- Toda relación en 2NF puede reducirse a un conjunto equivalente de relaciones 3NF
- El proceso es **reversible**, y por tanto que **no se pierde información** en la reducción
- La reducción a 3NF puede contener **información imposible de representar** en la relación en 2NF.

el hecho de que la situación de Roma es 50

Pérdida de Información

- ♦ Una **relación** con muchos atributos **mal diseñada** se puede **descomponer en dos ó más esquemas** con menos atributos.
- ♦ Si esta **descomposición** no se hace bien puede llegarse a **otra forma de diseño defectuoso**.

Si el esquema **PRIMERA** se descompone en dos esquemas:

A (S#, SITUACION, CIUDAD)

B (CIUDAD, P#, CANT)

Pérdida de Información

PRIMERA

S#	SITUACIÓN	CIUDAD	P#	CANT
S1	20	Londres	P1	300
S1	20	Londres	P2	200
S1	20	Londres	P3	400
S1	20	Londres	P4	200
S1	20	Londres	P5	100
S1	20	Londres	P6	100
S2	10	París	P1	300
S2	10	París	P2	400
S3	10	París	P2	200
S4	20	Londres	P2	200
S4	20	Londres	P4	300
S4	20	Londres	P5	400

Pérdida de Información

Se obtienen las relaciones

A (S#, SITUACION, CIUDAD)

S#	SITUACIÓN	CIUDAD
S1	20	Londres
S2	10	París
S3	10	París
S4	20	Londres

B (CIUDAD, P#, CANT)

CIUDAD	P#	CANT
Londres	P1	300
Londres	P2	200
Londres	P3	400
Londres	P4	200
Londres	P5	100
Londres	P6	100
París	P1	300
París	P2	400
París	P2	200
Londres	P4	300
Londres	P5	400

Si para alguna **consulta**
se necesita
reconstruir PRIMERA

A | x | B

La relación resultante es

S#	SITUACIÓN	CIUDAD	P#	CANT
S1	20	Londres	P1	300
S1	20	Londres	P2	200
S1	20	Londres	P3	400
S1	20	Londres	P4	200
S1	20	Londres	P5	100
S1	20	Londres	P6	100
S1	20	Londres	P4	300
S1	20	Londres	P5	400
S2	10	París	P1	300
S2	10	París	P2	400
S2	10	París	P2	200
S3	10	París	P1	300
S3	10	París	P2	400
S3	10	París	P2	200
S4	20	Londres	P1	300
S4	20	Londres	P2	200
S4	20	Londres	P3	400
S4	20	Londres	P4	200
S4	20	Londres	P5	100
S4	20	Londres	P6	100
S4	20	Londres	P4	300
S4	20	Londres	P5	400

Pérdida de Información

- ♦ Esta relación contiene **tuplas adicionales** respecto a PRIMERA.
- ♦ Las **consultas** que se efectúen podrían producir resultados erróneos
- ♦ Aunque se tienen más tuplas, **se pierde información**.
- ♦ Este tipo de descomposición se denomina **descomposición con pérdida** y es un **mal diseño**.

Es esencial que

al **descomponer una relación** en varias relaciones más pequeñas,
la **descomposición sea sin pérdida**.

La descomposición de **PRIMERA** en **SEGUNDA** y **SP**
es una **descomposición sin pérdidas**.



Pérdida de Información

- ♦ Una descomposición **sin pérdidas** garantiza que la **reunión** producirá exactamente la relación original.
- ♦ Una descomposición **con pérdidas** pierde información porque
 - la **reunión** puede producir un **superconjunto** de la relación original, y
 - no hay manera de saber cuáles tuplas son **espurias**.

Criterio para determinar si una descomposición tiene pérdida

Sean

R un esquema de relación

F un conjunto de DF en R.

R1 y **R2** una descomposición de R.

Esta **descomposición es sin pérdida** si por lo menos una de las siguientes DF está en F^+ :

$$R1 \cap R2 \rightarrow R1$$

$$R1 \cap R2 \rightarrow R2$$



Comentarios Finales

1- Los **objetivos generales del proceso de normalización** son:

- ♦ **Eliminar** ciertos tipos de **redundancia**
- ♦ **Evitar** ciertas **anomalías de actualización**
- ♦ Producir un **diseño** que sea una “**buen**a” **representación** del mundo real

Comentarios Finales

- 2- Para **solucionar los problemas de actualización**, se **descompone** una relación en **proyecciones** que estén en una forma normal adecuada.
- ♦ Sacar proyecciones para **eliminar** todas las **DF no completas**.
→ conjunto de relaciones **2NF**
 - ♦ Sacar proyecciones de una relación 2NF para **eliminar** las **DF transitivas**.
→ conjunto de relaciones **3NF**
 - ♦ Sacar proyecciones de las relaciones 3NF para eliminar **DF** restantes **en las cuales el determinante no sea una clave candidata**.
→ conjunto de relaciones **BCNF**.



Comentarios Finales



3- A veces hay razones válidas para **no normalizar por completo**.

Por ejemplo:

CALLE, CODIGOPOSTAL, CIUDAD y PROVINCIA

casi siempre se necesitan juntos y
los **códigos postales** no se modifican con mucha frecuencia,

tal descomposición implicará baja de performance.

En general:

datos **muy consultados** y **poco actualizados** no conviene **normalizar**.



Bibliografía



- ♦ Capítulo 21. Introducción a los Sistemas de Bases de Datos (Date)
- ♦ Capítulo 6. Fundamentos de Bases de Datos (Korth)