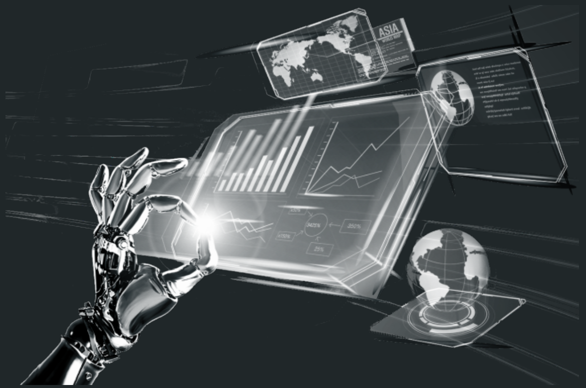


Gestión de Datos para Robótica

T1e - Normalización

Álvaro Vázquez Álvarez
Departamento de Electrónica e Computación

alvaro.vazquez@usc.es

Pabellón III - Despacho 4

Curso 2023-2024

Tabla de contenidos

Introducción

Fases del **diseño** de una **BD**

Modelo de Datos (MD)

MD Conceptual
Entidad-Relación

Independiente del SGBD

MD Lógico
Relacional
Red
Jerárquico

Dependiente del SGBD

MD Físicos
Propios de cada SGBD

Minimundo
↓
Análisis de Requisitos
↓
Diseño Conceptual
↓
Diseño Lógico
↓
Diseño Físico

Análisis de Requisitos:
- *Entrevistas con el cliente*

Diseño Conceptual:
- *Entidades*
- *Atributos*
- *Relaciones*

Diseño Lógico:
- *Tablas, relaciones (relacional)*
- *Grafos (Red)*
- *Árboles (Jerárquico)*

SGBD

Diseño Físico:
- *Punteros*
- *Índices*
- *Diccionarios*
- *....*

Introducción

En grandes esquemas de relación, pueden aparecer los siguientes problemas cuando el diseño del esquema relacional no es adecuado:

- **Incapacidad** para almacenar ciertos hechos
- **Redundancias**, y por tanto, posibilidad de **inconsistencias**
- **Aparición** en la base de datos de **estados inválidos** en el mundo real:
  - Anomalías de inserción → imposibilidad de dar de alta una tupla por no disponer del valor de un atributo principal
  - Anomalías de borrado → pérdida de información por dar de baja una tupla
  - Anomalías de modificación → tiene que ver con la redundancia (repetición de la misma información en tuplas diferentes y consiguiente necesidad de propagar actualizaciones)

Introducción

Ejemplo de Diseño inadecuado

ESCRIBE					
AUTOR	NACIONALIDAD	COD_LIBRO	TITULO	EDITORIAL	AÑO
Date, C.	Norteamericana	23433	Databases	Adisson-W.	2008
Date, C.	Norteamericana	23433	SQL Standard	Adisson-W.	2006
Date, C.	Norteamericana	23433	Guide To DB	Adisson-W.	2008
Codd, E.	Norteamericana	97875	Relational M.	Adisson-W.	2013
Gardarin	Francesa	34245	Base de Datos	Paraninfo	2006
Gardarin	Francesa	86754	Comparación BD	Eyrolles	2014
Valduriez	Francesa	86754	Comparación BD	Eyrolles	2014
Kim, W.	Norteamericana	23456	OO Databases	ACM Press	2009
Lochovsky	Canadiene	23456	OO Databases	ACM Press	2009

Problemas

La nacionalidad se repite por cada autor y libro

La editorial y año se repiten con libros que tienen múltiples autores

Introducción

Ejemplo de Diseño inadecuado

ESCRIBE					
AUTOR	NACIONALIDAD	COD_LIBRO	TITULO	EDITORIAL	AÑO
Date, C.	Norteamericana	23433	Databases	Adisson-W.	2008
Date, C.	Norteamericana	23433	SQL Standard	Adisson-W.	2006
Date, C.	Norteamericana	23433	Guide To DB	Adisson-W.	2008
Codd, E.	Norteamericana	97875	Relational M.	Adisson-W.	2013
Gardarin	Francesa	34245	Base de Datos	Paraninfo	2006
Gardarin	Francesa	86754	Comparación BD	Eyrolles	2014
Valduriez	Francesa	86754	Comparación BD	Eyrolles	2014
Kim, W.	Norteamericana	23456	OO Databases	ACM Press	2009
Lochovsky	Canadiene	23456	OO Databases	ACM Press	2009

Problemas

Anomalías de inserción: al dar de alta un libro es preciso insertar tantas tuplas como autores tenga el libro

Anomalías de modificación: al cambiar la editorial un libro es preciso modificar todas la tuplas que corresponden a ese libro

Anomalías de borrado: borrar un libro obliga a borrar varias tuplas, tantas como autores tenga ese libro y, viceversa, el borrado de un autor lleva a borrar tantas tuplas como libros ha escrito ese autor

Imposibilidad de almacenamiento de ciertos datos (hechos):

- No es posible introducir obras anónimas
- No se puede almacenar información sobre autores sin libro (no tendría cod_libro que es PK).

Desaparición de informacion:

- Al dar de baja un libro se pierde la información de los autores si sólo tienen ese libro en la BD.

Introducción

Ejemplo de Diseño inadecuado

- ¿Como se debería haber diseñado la relación anterior?

libro (cod_libro, titulo, editorial, año)
autor (nombre, nacionalidad)
escribe (cod_libro, nombre)

Diseño Conceptual (Modelo Entidad-Relación)

Diseño Lógico (Modelo Relacional)

Conclusión:
Si se realiza un buen diseño conceptual (Modelo E-R) y una correcta transformación al Modelo Relacional se evitan gran parte de los problemas comentados anteriormente.

Existen tres medidas informales de calidad para el diseño de esquemas de relación:

- Reducción de información redundante en las tuplas.
- Reducción de valores nulos.
- Eliminación de la posibilidad de obtención de tuplas espurias (tuplas con información incompleta)

Dependencias Funcionales (DF)

Toda relación puede definirse de dos formas:

- **Por intensión:** especificando el esquema de la relación.
- **Por extensión:** especificando todas y cada una de las tuplas que componen la relación.

AUTOR	NOMBRE	NACIONALIDAD
	Date, C.	Norteamericana
	Date, C.	Norteamericana
	Date, C.	Norteamericana
	Codd, E.	Norteamericana
	Gardarin	Francesa
	Gardarin	Francesa
	Valduriez	Francesa
	Kim, W.	Norteamericana
	Lochovsky	Canadiene

Por intensión

Por extensión

Dependencias Funcionales (DF)

Esquema de relación: estructura abstracta que **define** una **relación** a **través** de un **nombre**, un **conjunto de atributos y** un **conjunto de restricciones** que caracterizan a esa relación.

- Esquema de relación → r = R(T,L) donde:
  - R es el nombre de la relación.
    - Ejemplo: R = libro.
  - T es el conjunto de atributos que definen a R.
    - Ejemplo: T = {cod_libro, titulo, editorial, año}.
  - L es el conjunto de restricciones que caracterizan a R.
    - Las restricciones que permiten definir la semántica del problema (significado de los datos en función de sus relaciones con otros), se conocen como **dependencias funcionales**.
    - Ejemplo $L = \{L_1, L_2, \dots L_n\}$
- Descriptor: subconjunto de T → subconjunto de atributos que definen a R.

Dependencias Funcionales (DF)

Dependencia Funcional (DF)

- Es una restricción entre dos conjuntos de atributos (descriptores) de una relación.
- Dada una relación $R(A_1, A_2, \ldots, A_n)$ y siendo X e Y descriptores de la relación R, se dice que Y es funcionalmente dependiente de X ($X \rightarrow Y$), si cada valor de X determina unívocamente el valor de Y
  - Es decir:
    - Si dos tuplas son iguales para los atributos X, también lo son para los Y
    - A cada valor de X le corresponde un ÚNICO valor de Y
- **No** puede **deducirse** de la **extensión** de la relación, debe ser **conceptual**.
- **$X \rightarrow Y$ no implica** obligatoriamente que **$Y \rightarrow X$**
  - Si **$X \rightarrow Y \land Y \rightarrow X$ entonces X** e **Y** son ***descriptores equivalentes***.
- Especifican restricciones sobre los atributos que deben cumplirse en todo momento (invariantes en el tiempo)
- Si X es una clave candidata de R $\Rightarrow X \rightarrow Y \ \forall Y$ de R

`Todo atributo que es clave en la relación tiene una dependencia funcional con el resto de atributos de la relación`

Dependencias Funcionales (DF)

Dependencia Funcional (DF) :

- Otra forma de verlo:

Una dependencia funcional determina una relación N:1 entre dos conjuntos de atributos ($X \xrightarrow{N:1} Y$)

- Para un valor de X, sólo puede haber un valor de Y.
- Para un valor de Y, habrá en general varios valores de X.

Dependencias Funcionales (DF)

Dependencia Funcional (DF)

- Ejemplo (I): data el esquema de relación **coche**(matricula, marca, modelo, potencia, motor)

COCHE				
MATRICULA	MARCA	MODELO	POTENCIA	COLOR
1455GHV	SEAT	ALTEA 1.9 TDI	90	AZUL
6551DTG	SEAT	ALTEA 1.9 TDI	90	ROJO
1442HTG	AUDI	A3	105	NEGRO

Se podría extraer el siguiente conjunto de DF:

L = {matrícula → color, modelo → potencia, modelo → marca, matrícula → modelo}

Basándose solamente en las tuplas de la tabla, ¿existen otras DF's?

modelo → color ⇒ NO!!. Porque hay dos modelos de coche iguales asociados a diferentes colores

potencia → matrícula ⇒ NO!!. Porque hay coches con potencias iguales asociados a diferentes matriculas

marca → modelo ⇒ SI!!. Los coches de la misma marca tienen asociado el mismo modelo.

Dependencias Funcionales (DF)

Dependencia Funcional (DF)

- Ejemplo (II): dada la relación **persona**(dni, calle, mun, prov, cp). Se cumplen las siguientes restricciones semánticas:
  a. dni es el identificador de una persona.
  b. Una persona sólo vive en una calle, un municipio, una provincia y un cp.
  c. Un municipio pertenece a una provincia y no existen dos municipios con el mismo nombre.
  d. No existen cps que incluyan más de un municipio, pero un municipio puede incluir más de un cp.
  e. Una calle de un municipio de una provincia pertenece a un único cp.
- Identificar el conjunto *L* de DF que se obtienen de las restricciones:

L = {dni → calle, mun, prov, cp,
mun → prov,
cp → mun
calle,num,prov → cp}

¿Hay alguna otra?

Dependencias Funcionales (DF)

Dependencia Funcional (DF)

- Ejercicio (II): dada la relación **empleados_centro**(idEmp, nombreE, direcciónE, puestoE, salarioE,centro, direcciónC, teléfonoC). Se cumplen las siguientes restricciones semánticas:
    a. Cada empleado tiene un único identificador (idEmp).
    b. Cada puesto está asociado a un único centro, pero en un centro puede haber varios puestos.
    c. La dirección y el teléfono son únicos para cada centro.
- Identificar el conjunto *L* de DF que se obtienen de las restricciones

Dependencias Funcionales (DF)

Tipos de Dependencias funcionales:

- Una DF $X \rightarrow Y$ es **parcial** cuando:
  - Y es funcionalmente dependiente de X
  - X es solo una parte de la clave de la tabla.

    $AB \rightarrow C$
    $B \rightarrow C$ ⇒ $AB \rightarrow C$ porque si elimino A, se sigue cumpliendo la DF (A sobra)
- Una DF $X \rightarrow Y$ es **total** cuando:
  - No es una DF parcial

    $A \rightarrow C$ ⇒ $A \rightarrow C$ no existe un subconjunto de A que determine C
- Una DF $X \rightarrow Y$ es **trivial** cuando:
  - Y es un subconjunto de X

    $AB \rightarrow B$ ⇒ B es un subconjunto de AB
- Una DF $X \rightarrow Y$ es **elemental** cuando:
  - Y es un **atributo único**
  - Es una DF **total** y **no trivial** ($Y \not\subset X$)

Dependencias Funcionales (DF)

Reglas de inferencia: siendo X, Y, Z conjuntos de atributos

- Reglas de Armstrong:
  - **Reflexividad**: si $Y \subseteq X$, entonces $X \rightarrow Y$
    - Ejemplo: $A \rightarrow A$
  - **Aumento**: $X \rightarrow Y \Rightarrow ZX \rightarrow ZY$
    - Ejemplo: $B \rightarrow D \Rightarrow BC \rightarrow DC$
  - **Transitividad**: $X \rightarrow Y, Y \rightarrow Z \Rightarrow X \rightarrow Z$
    - Ejemplo: $A \rightarrow BC, BC \rightarrow DC \Rightarrow A \rightarrow DC$
- Reglas de inferencia:
  - **Descomposición**: $X \rightarrow YZ \Rightarrow X \rightarrow Y$ y $X \rightarrow Z$
    - Ejemplo: $A \rightarrow BC \Rightarrow A \rightarrow B$ y $A \rightarrow C$
  - **Unión**: $X \rightarrow Y, X \rightarrow Z \Rightarrow X \rightarrow YZ$
    - Ejemplo: $A \rightarrow A, A \rightarrow B \Rightarrow A \rightarrow AB$
  - **Pseudo-transitividad**: $X \rightarrow Y, WY \rightarrow Z \Rightarrow WX \rightarrow Z$
    - Ejemplo: $A \rightarrow C$ y $CD \rightarrow E \Rightarrow AD \rightarrow E$
- Es decir $L^+ = L \cup$ {DF's inferidas mediante los axiomas de Armstrong} $\Rightarrow L \subseteq L^+$

$\boldsymbol{R}(A, B, C, D, E)$

$L = \{A \rightarrow BC, CD \rightarrow E, B \rightarrow D, E \rightarrow A\}$

Dependencias Funcionales (DF)

Reglas de inferencia:

- Ejemplo:
  - Dado el esquema de relación: ***R***(A, B, C, D, E; {A ➞ B, C ➞ D, D ➞ E})
  - Demostrar, aplicando los axiomas de Armstrong que: **AC➞ABCDE**
    - Pasos*:
      1. A ➞ B (dada)
      2. AC ➞ ABC (aumento con C)
      3. C ➞ D (dada)
      4. D ➞ E (dada)
      5. C ➞ E (transitividad con 3, 4)
      6. C ➞ DE (unión con 3, 5)
      7. ABC ➞ ABDE (aumento con AB)
      8. ABC ➞ ABCDE (reflexividad con C)
      9. AC ➞ ABCDE (transitividad con 2, 8)

*existen múltiples formas de llegar al mismo resultado

Dependencias Funcionales (DF)

Cierre transitivo ($\boldsymbol{L}^+$):

- Conocer ciertas DF permite inferir la existencia de DF ocultas:
  - Ejemplo: sea **coche**(T, L)
    - Donde:

      T = {matricula, modelo, marca, potencia, color}

      L = {matricula → modelo; modelo → marca; modelo → potencia; matricula → color}
    - Sabiendo que *matricula* → *modelo* y que *modelo* → *marca*. ¿Se podría inferir alguna DF que no está en el conjunto L?

      ```
      matricula → marca
      ```
    - ¿Se puede deducir alguna otra DF?

      ```
      matricula → potencia
      ```

- El conjunto de DF que son consecuencia lógica del conjunto inicial L de DF se le denomina cierre transitivo de L y se representa por $\boldsymbol{L}^+$

$$\boldsymbol{L}^+ = L \cup \{\text{matricula} \rightarrow \text{marca, matricula} \rightarrow \text{potencia}\}$$

Dependencias Funcionales (DF)

Cierre transitivo (L^+):

Conceptualmente:
Un cierre transitivo es la unión de las dependencias funcionales inherentes (propias) a la relación más las que se pueden obtener tras aplicar los axiomas de armstrong junto con las reglas de inferencias.

Cierre de un descriptor respecto de *L*

Inferir todas las DF es un proceso muy complejo ya que hay múltiples combinaciones posibles

Inferir todas las DF es un proceso muy complejo ya que hay múltiples combinaciones posibles ➡ es fácil saltarse DF.

$$\frac{n!}{(n-r)!(r)!}$$

r= número de atributos
n=número de atributos para la clave

	1	2	3	4	5	
Posibles claves	A	AB	ABC	ABCD	ABCD E	
	B	AC	ABD	ABCE		
	C	AD	ABE	ABCF		
	D	...	...	...		
	E	DE	CDE	BCDE		
Total	5	10	10	5	1	31

Asumamos que tenemos una relación con 5. Se podrían dar hasta 31 combinaciones

Cierre de un descriptor respecto de *L*

El cierre de un descriptor respecto de *L* permite obtener todas las dependencias funcionales de manera sistemática.

- Determina el conjunto de atributos X del lado izquierdo de alguna DF en *L*
- Determinar el conjunto de todos los atributos que son dependientes de X (lado derecho).

Algoritmo:

ENTRADA:
- Un conjunto de DFs y atributos *R*(T,*L*)
- Un descriptor X, subconjunto de T.

SALIDA:
- $X^+ = X$
- Repetir hasta que X^+ no cambie más.
  - Para cada DF $(Y \rightarrow Z) \in L$ tal que $Y \subseteq X^+$

    $X^+ = X^+ \cup Z$

Cierre de un descriptor respecto de *L*

Ejemplo(I): sea el esquema de relación R(T, L),donde:

$T= \{A, B, C, D, E, F\}$

$L= \{AB \rightarrow C, BC \rightarrow AD, D \rightarrow E, CF \rightarrow B\}$

Calcular el cierre del descriptor AB respecto de L:

$(AB)^+ = AB$

$(AB)^+ = ABC$ (ya que $AB \rightarrow C$ y $AB \subset X^+$ y)

$(AB)^+ = ABCD$ (ya que $BC \rightarrow AD$ y $BC \subset X^+$)

$(AB)^+ = ABCDE$ (ya que $D \rightarrow E$ y $D \subset X^+$)

Ejemplo(II): sea el esquema de relación R(T, L),donde:

$T= \{A, B, C, D, E, F\}$

$L= \{AB \rightarrow C, C \rightarrow A, BC \rightarrow D, ACD \rightarrow B, D \rightarrow EF, BE \rightarrow C, CF \rightarrow BD, CE \rightarrow AF\}$

Calcular el cierre del descriptor BD respecto de L:

$(BD)^+= BD$

$(BD)^+= BDEF$ (ya que $D \rightarrow EF$ y $D \subset X^+$)

$(BD)^+= BDEFC$ (ya que $BE \rightarrow C$ y $B \subset EX^+$)

$(BD)^+= ABDEFC$ (ya que $C \rightarrow A$ y $C \subset X^+$)

Recubrimiento

Para definir un recubrimiento no redundante (o mínimo) hay que definir dos conceptos:

- **Recubrimiento** de un **conjunto** de **dependencias funcionales**: dados dos conjuntos de DFs S_1 y S_2, se dice que $\mathbf{S_2}$ **es** un **recubrimiento** de $\mathbf{S_1}$ **si cada dependencia de** $\mathbf{S_1}$ **se deduce de** $\mathbf{S_2}$ (es decir, se puede demostrar que cada dependencia de S_1 está en el cierre de S_2).
- **Equivalencia entre conjuntos** de **dependencias funcionales**: dos conjuntos de dependencias funcionales $\mathbf{S_1}$ **y** $\mathbf{S_2}$ **son equivalentes si** $\mathbf{S_1^+ = S_2^+}$.
  - Es decir: S_1 es un recubrimiento de S_2 y S_2 es un recubrimiento de S_1
- **Conjunto mínimo:** es un conjunto de dependencias que cumple:
  - **Todas** las DFs son **elementales** → ($X \rightarrow Y$, Y es único y X es irreducible)
  - **No** existen DF **redundantes**
- **Recubrimiento no redundante**: es el conjunto mínimo de DF que es equivalente a L^+
  - Dado un conjunto de DF, siempre se podrá calcular al menos un r.nr.

Recubrimiento: si cada dependencia de S_1 se deduce en S_2 → $S_1^+ = S_2^+$

Recubrimiento

Ejemplo: dados los siguientes conjuntos de DFs comprobar si son equivalentes ($L^+ = M^+$):

$$\boldsymbol{L} = \{A \rightarrow B, B \rightarrow A, A \rightarrow C, A \rightarrow D\}$$
$$\boldsymbol{M} = \{A \rightarrow B, B \rightarrow A, B \rightarrow C, B \rightarrow D\}$$

<u>*M* es recubrimiento de *L*.</u> Las dependencias A ➞ B y B ➞ A están en ambos conjuntos, por lo que las únicas dependencias de *L* que no están en *M* son A ➞ C y A ➞ D.

- Hacer el cierre de *M* para los atributos de *L* que no están en *M* (A ➞ C y A ➞ D)
  - A^+ respecto de M = {ABCD}, como C y D ∈ A^+ entonces las dependencias de *L* están en M^+. (*M* es un recubrimiento de *L*)

<u>*L* es recubrimiento de *M*.</u> las dependencias A ➞ B y B ➞ A están en ambos conjuntos, por lo que las únicas dependencias de *L* que no están en *M* son B ➞ C y B ➞ D.

- Hacer el cierre de *L* para los atributos de *M* que no están en *M* (B ➞ C y B ➞ D)
  - B^+ respecto de L = {BACD}, como C y D ∈ B^+ entonces las dependencias de *M* están en L^+. (*L* es un recubrimiento de *M*)

$$\boxed{L^+ = M^+}$$

Recubrimiento no redundante

Conceptualmente:
R.N.R: Conjunto de DFs funcionales mínimas (irreducibles) que aseguran que se cumple L^+.

Propiedades de un R.N.R: dado *S* un conjunto de dependencias funcionales:

- La parte derecha de cada dependencia funcional de *S* tiene sólo un atributo.
- La parte izquierda de cada dependencia funcional es irreducible en el sentido en que si se elimina algún atributo, necesariamente cambia el cierre de *S*.
- No se puede eliminar ninguna dependencia funcional de *S* sin cambiar su cierre.

→ **Definición de Conjunto Mínimo**

Recubrimiento no redundante

Algoritmo:

ENTRADA:

- Conjunto de DFs L

SALIDA:

- Recubrimiento no redundante M

PROCESO:

- Paso 1: segundos miembros simples. Para toda dependencia $X \rightarrow Y$ de L se sustituye por $X \rightarrow A_1$, $X \rightarrow A_2$, ..., $X \rightarrow A_k$, siendo $A_1, A_2, \ldots, A_k$ atributos de Y. El conjunto resultante se llama $L^{(1)}$

  $X \rightarrow YZ \Rightarrow X \rightarrow Y$ y $X \rightarrow Z$

- Paso 2: Eliminación de atributos extraños. Para toda dependencia $X \rightarrow A_i$ de $L^{(1)}$, si $B_i \in X$ y siendo $Z=X-\{B_i\}$ se calcula Z^+ respecto de $L^{(1)}$.
  - Si $A_i \in Z^+$, entonces $(Z \rightarrow A_i) \in L^{(1)+}$, de modo que la sustitución de $X \rightarrow A_i$ por $Z \rightarrow A_i$ conduce a un conjunto equivalente.

    Si eliminando un atributo de la izquierda obtengo el mismo cierre entonces sobra
  - Al conjunto resultante se le llama $L^{(2)}$
- Eliminar dependencias redundantes. Para cada $X \rightarrow Y \in L^{(2)}$, ver si $(X \rightarrow Y) \in (S' - \{X \rightarrow Y\})^+$, es decir si en $(L^{(2)} - \{X \rightarrow Y\})$ se cumple $Y \in X^+$

  Si eliminando una DF $X \rightarrow Y$ y hago el cierre y sigo obteniendo $Y \in X$ entonces sobra

Recubrimiento no redundante

Ejemplo: Sea el esquema $R(T=\{A, B, C\}, L= \{A \rightarrow BC, B \rightarrow C, A \rightarrow B, AB \rightarrow C\})$

Paso 1: segundos miembros simples.

$A \rightarrow BC$ se descompone en $A \rightarrow B$ y $A \rightarrow C$, pero $A \rightarrow B$ ya existe. Se añade sólo $A \rightarrow C$.

$\mathbf{L^{(1)} = \{A \rightarrow B, A \rightarrow C, B \rightarrow C, AB \rightarrow C\})}$

Paso 2: eliminación de atributos extraños (parte izquierda).

$AB \rightarrow C \in L^{(1)}$

¿A es extraño? $\Rightarrow$ $\{B\}^+$ respecto de $L^{(1)} \rightarrow \{BC\}$, $C \in L^{(1)+}$ entonces el atributo A es extraño.

$AB \rightarrow C$ se transforma en $B \rightarrow C$. Pero la DF $B \rightarrow C$ ya existe en $L^{(1)}$ así que no se incluye en $L^{(2)}$.

$\mathbf{L^{(2)} = \{A \rightarrow B, A \rightarrow C, B \rightarrow C\}}$

Paso 3: eliminación de dependencias redundantes.

$A \rightarrow B$: $(A)^+_{\{L\text{-}A \rightarrow B\}} = \{AC\} \Rightarrow B \notin \{AC\}$, por tanto $A \rightarrow B$ no es redundante.

$A \rightarrow C$: $(A)^+_{\{L\text{-}A \rightarrow C\}} = \{ABC\} \Rightarrow C \in \{ABC\}$, por tanto $A \rightarrow C$ es redundante.

$L^{(3)} = \{A \rightarrow B, \cancel{A \rightarrow C}, B \rightarrow C\}$

$B \rightarrow C$: $(B)^+_{\{L\text{-}B \rightarrow C\}} = \{B\} \Rightarrow C \notin \{B\}$, por tanto $B \rightarrow C$ no es redundante.

$\mathbf{L^{(3)} = \{A \rightarrow B, B \rightarrow C\}}$

Superclave y Clave Candidata

Superclave: se dice que un descriptor S_K, subconjunto propio del conjunto de atributos del esquema ($S_K \subseteq T$), es **superclave** del esquema ***R***(T,*L*) cuando se cumple $S_K \rightarrow T$, es decir, $(S_K \rightarrow T) \in L^+$.

- Ejemplo: dado el esquema por extensión ***R***(T,L) donde:

  T={A,B,C,D,E}

  L={A ➞ B, C ➞ D, D ➞ E}

  - AC es superclave?. ¿Es decir, AC ➞ ABCDE ∈ L^+?
  - ABC es superclave? ¿Es decir, ABC ➞ ABCDE ∈ L^+?
  - E es superclave? ¿Es decir, E ➞ ABCDE ∈ L^+?

Una superclave: un atributo (o conjunto de atributos) que aseguran que los valores de las tuplas son únicos

Clave candidata: se dice que un descriptor K, subconjunto propio del conjunto de atributos del esquema ($K \subseteq T$), es **clave** del esquema ***R***(T,*L*) **si**, **además de ser superclave**, **no** existe **ningún subconjunto** estricto de K **con la misma propiedad**, es decir, $(K \Rightarrow T) L^+$.

- Ejemplo: dado el esquema por extensión ***R***(T,*L*) donde:

  T={A,B,C,D,E}

  L={A ➞ B, C ➞ D, D ➞ E}

  - AC es clave?. ¿Es decir, AC ➞ ABCDE ∈ L^+?
  - ABC es clave? ¿Es decir, ABC ➞ ABCDE ∈ L^+?
  - E es clave? ¿Es decir, E ➞ ABCDE ∈ L^+?

Una clave: una superclave cuyos atributos son mínimos (todos son necesarios para asegurar la unicidad de las tuplas)

Superclave y Clave Candidata

Algoritmo:

ENTRADA:

- Una relación $R<T, L>$

SALIDA:

- Conjunto de claves candidatas C_{Ks}

PROCESO:

- Paso 1: Calcular el RNR de L (L_{min}) e inicializar el conjunto de atributos de cálculo, $C_a = A_1, \ldots, A_n$
- Paso 2: Eliminar atributos independientes (A_{indep}). Son los atributos que no están presentes en ninguna dependencia. $C_a = C_a - A_{indep}$
- Paso 3: Hallar términos equivalentes (A_{equiv}). Pares (X, Y) de términos que cumplen $X \rightarrow Y$ e $Y \rightarrow X$ con $X \cap Y = \{\emptyset\}$. Si existen dejar solo un atributo, eliminar los restantes de C_a y calcular la proyección de L_{min} en C_a, obteniendo L_c
- Paso 4: Construcción claves tentativas K con todos los elementos que sean implicantes (izquierda) calcular el cierre (K^+). Si $K^+ = C_a$ entonces K es clave. Si $K^+ \neq C_a$ entonces se van añadiendo otros atributos (subconjuntos de 1 elemento, luego de 2, ...).
- Paso 5: Por cada K encontrado como clave de C_a se unen los atributos independientes, A_{indep}, y se agrega K al resultado, C_{Ks}.
- Paso 6: Repetir paso 4 y 5 con términos equivalentes encontrados en el paso 3.

Superclave y Clave Candidata

Ejemplo: dada la relación R(A, B, C, D, E, F, G) y el conjunto de dependencias funcionales, L = {AB→F, D→A, E→D, D→E, CF→B, B→C}, calcule todas las claves candidatas existentes.

Paso 1: L ya es RNR ⇒ L_{min}= {AB→F, D→A, E→D, D→E, CF→B, B→C}.

Paso 2: A_{indep} = {G}, C_a= {A, B, C, D, E, F}

Paso 3: E→D, D→E y E∩D={Ø} ⇒ eliminamos D ⇒ C_a= {A, B, C, E, F}.

⇒ L_c = {AB→F, **E→A**, CF→B, B→C}

Si elimino D tengo que quitar las DFs que lo contienen. Si tengo que D→A, E→D, D→E entonces puedo asumir por transitividad que tengo E→A

Paso 4: K={E}, pero $(E)^+$ = {A, E} ≠ C_a ⇒ E no es clave.

Paso 5: Agregamos otros implicantes.

Grupos de un atributo.

$(EA)^+$ = {A, E} ≠ C_a ⇒ EA no es clave.

$(EB)^+$ = {A, B, C, E, F} = C_a ⇒ EB es clave.

$(EF)^+$ = {A, E, F} ≠ C_a ⇒ EF no es clave.

Grupos de dos atributos.

$(EAC)^+$ = {A, C, E} ≠ C_a ⇒ EAC no es clave.

$(EAF)^+$ = {A, E, F} ≠ C_a ⇒ EB no es clave

$(ECF)^+$ = {A, B, C, E, F} = C_a ⇒ EF es clave.

¡¡ NO PRUEBO MÁS COMBINACIONES PORQUE EL RESTO INCLUYE ATRIBUTOS QUE YA SON CLAVE !!
EBC ⊃ EB
ECFA ⊃ ECF
....

Superclave y Clave Candidata

Ejemplo: dada la relación R(A, B, C, D, E, F, G) y el conjunto de dependencias funcionales, $L = \{AB \rightarrow F, D \rightarrow A, E \rightarrow D, D \rightarrow E, CF \rightarrow B, B \rightarrow C\}$, calcule todas las claves candidatas existentes.

Paso 6: repetir pasos 4 y 5 con términos equivalentes:

$D \leftrightarrow E$ (equivalentes)

Sustituir E por D en los atributos clave encontrados en los pasos 4 y 5.

Paso 4:

E no es clave $\Rightarrow$ D no es clave.

Paso 5:

EB es clave $\Rightarrow$ DB es clave

ECF es clave $\Rightarrow$ DCF es clave.

$K_s = \{EB, DB, ECF, DCF\}$

Formas Normales

Teoría de la Normalización:

- Se basa en la descomposición de una relación en subrelaciones equivalentes a la original (preservan dependencias y verifican join sin pérdida).
- Se dice que un esquema está en una determinada forma normal (1FN, 2FN, 3FN, FNBC, 4FN, 5FN) si satisface un conjunto de propiedades.
    - La tres primeras formas normales (1FN, 2FN, 3FN) permiten realizar una **descomposición sin pérdida** a partir del concepto de DF.
        - Preservación de atributos: $R = \cup_{(i=1,k)} R_i$
        - Join sin pérdida: $\Pi_{R'}(r) \infty \Pi_{R''}(r) \infty \ldots \infty \Pi_{R''''}(r) = r, \ \forall r$
        - Preservación de dependencias: $L^+ = \{\cup_{(i=1,k)} L_i\}^+$

<u>Preservación de atributos</u>
Si uno los atributos de las subrelaciones obtengo los atributos de la relación original

<u>Join sin pérdida</u>
Si hago un join de todas las subrelaciones no pierdo información (tuplas) de la relación original

<u>Preservacipon de dependencias</u>
Si hago la unión de los cierres de las subrelaciones obtengo el cierre de la relación original

Formas Normales

Primera Forma Normal: Una relación *R* está en **1FN** si **todos** los **atributos** contienen un único **valor atómico** $\Rightarrow$ no se permiten atributos multivaluados.

- Es una restricción inherente al propio modelo relacional.
- Proceso de normalización: dos opciones
  - Opción 1: crear una tupla para cada valor diferente $\Rightarrow$ (clave = clave + atributo)
  - Opción 2: crear una nueva relación con los diferentes valores del atributo.

LIBRO		
<u>COD_LIBRO</u>	TITULO	AUTOR
654654	Data models	Tsichiritzis Lochovosky
665465	A guide to DB2	Date
876545	Bases de Datos	Gardarin Valduriez

Opción 1 $\Rightarrow$

LIBRO		
<u>COD_LIBRO</u>	TITULO	<u>AUTOR</u>
654654	Data models	Tsichiritzis
654654	Data models	Lochovosky
665465	A guide to DB2	Date
876545	Bases de Datos	Gardarin
876545	Bases de Datos	Valduriez

¿Porque hay que poner una clave compuesta?

Formas Normales

Primera Forma Normal: Una relación *R* está en **1FN** si **todos** los **atributos** contienen un único **valor atómico** $\Rightarrow$ no se permiten atributos multivaluados.

- Es una restricción inherente al propio modelo relacional.
- Proceso de normalización: dos opciones
  - Opción 1: crear una tupla para cada valor diferente $\Rightarrow$ (clave = clave + atributo)
  - Opción 2: crear una nueva relación con los diferentes valores del atributo.

LIBRO		
COD_LIBRO	TITULO	AUTOR
654654	Data models	Tsichiritzis Lochovosky
665465	A guide to DB2	Date
876545	Bases de Datos	Gardarin Valduriez

Opción 2 ⇒

LIBRO	
COD_LIBRO	TITULO
654654	Data models
665465	A guide to DB2
876545	Bases de Datos

AUTOR	
COD_LIBRO	AUTOR
654654	Tsichiritzis
654654	Lochovosky
665465	Date
876545	Gardarin
876545	Valduriez

```
¿Qué opción es mejor? ¿Porqué?
```

Formas Normales

Segunda Forma Normal: Una relación R está en **2FN si cada atributo** A de R **cumple al menos** una de las siguientes condiciones:

- A aparece en una clave candidata
- A no es parcialmente dependiente de una clave candidata ($A \subset PK$) $\Rightarrow$ DF no completa

SEGUNDA FORMA NORMAL
Si está en **1FN** y cada **atributo** que **no** es **clave depende** de la **clave primaria completa**.

IMPLICACIÓN
Si un **esquema** R está en **1FN** y su **clave primaria** es **simple** (un solo atributo) **entonces siempre** va a estar en **2FN**.

- Proceso de normalización:
  - Crear tantas nuevas relaciones como dependencias funcionales sean no completas

Formas Normales

Segunda Forma Normal: Una relación *R* está en **2FN si cada atributo** A de *R* **cumple al menos** una de las siguientes condiciones:

- Proceso de normalización:
  - Crear tantas nuevas relaciones como dependencias funcionales sean no completas.
- Ejemplo:

suministrador(proveedor, articulo, direccion, precio)
L={proveedor, articulo → precio; proveedor → dirección}
K={proveedor, articulo}

¿Está en 2FN? ⇒ NO!. Existe un atributo no clave (direccion) que NO depende totalmente de la clave

1. Descomponer y **crear una nueva relación para cada parte** de la **clave** con su **atributo** o atributos **dependientes**.
2. **Crear** otra **relación** con las claves originales y los atributos que dependan funcionalmente **de manera total** de ellas.

⇒ **suministrador**(proveedor, direccion) *K*={proveedor}
⇒ **producto**(proveedor, articulo, precio) *K*={proveedor, artículo}

Formas Normales

Tercera Forma Normal: Una relación *R* está en **3FN si está** en **2FN** y **no contiene atributos no clave** con **dependencias transitivas**.

- Proceso de normalización:
  1. Separar las dependencias funcionales que se contraponen con la **3FN** creando nuevas relaciones R_i.
  2. En la relación original, eliminar los atributos dependientes (R_1).
  3. Si ninguna relación R_i contiene una clave de R, crear una relación adicional que contenga atributos que formen una clave de R
- Ejemplo:

coche(nm, marca, tipo, potencia, color)

L={nm→color; tipo→potencia; tipo→marca; nm→tipo}

K={<u>nm</u>}

¿Está en 3FN?

NO!. Hay atributos no clave (marca, potencia) que dependen de otro atributo no clave (tipo) a través de tipo→marca y tipo→potencia. Es decir, existe las relaciones transitiva nm→potencia y nm→marca a través de tipo que no es clave.

R

$\boldsymbol{R_1}$(nm, color, tipo), $\boldsymbol{L_1}$ = {nm→color, nm→tipo}

$\boldsymbol{R_2}$(tipo, potencia, marca), $\boldsymbol{L_2}$ = {tipo→potencia, tipo→marca}

Formas Normales

Forma Normal de Boyce-Codd: Una relación *R* está en **FNBC** cuando toda DF X→Y, **X es clave o superclave**.

- Proceso de normalización:
  1. Si X → A es una DF de R que viola la FNBC, es decir, X no es superclave ⇒ descomponer la relación R en:
     - $\boldsymbol{R_1}$ tal que T_1={X, A}
     - $\boldsymbol{R_2}$ tal que T_2= T-{A}
  2. Si $\boldsymbol{R_2}$ no está en FNBC aplicar recursivamente el paso 1
- Ejemplo:

  asiste(cod_curso, nom_curso, cod_estudiante, nota)

  L={cod_curso, cod_estudiante→nota, nom_curso→cod_curso, cod_curso→nom_curso}

  K={<u>(cod_curso,cod_estudiante), (nom_curso, cod_estudiante)</u>}

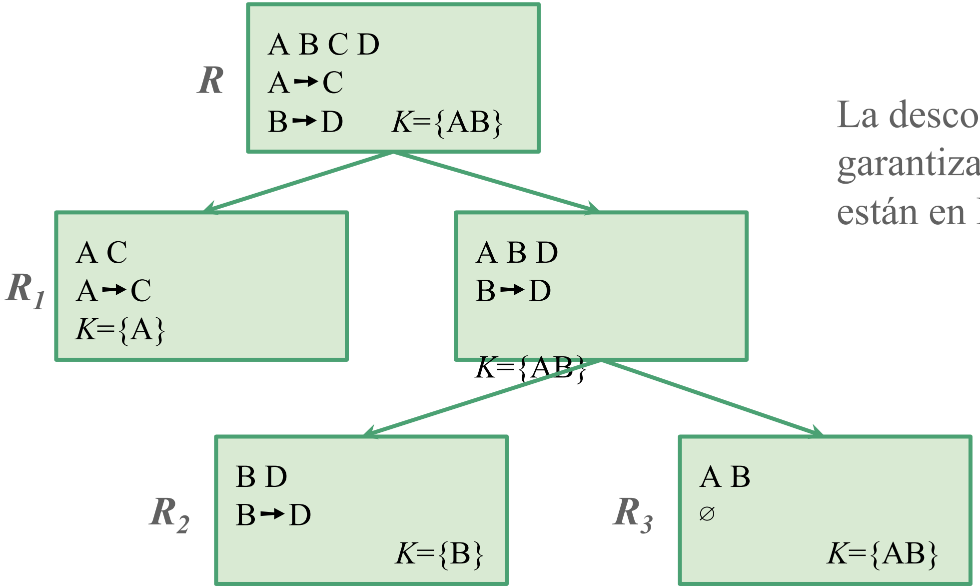
¿Está en FNBC?

NO!. Hay DFs con implicantes que no son clave ni superclave (nom_curso→cod_curso, cod_curso→nom_curso)

Formas Normales

Forma Normal de Boyce-Codd: una relación *R* está en **FNBC** cuando toda DF X ➞ Y, **X es clave o superclave**.

- Ejemplo: normalizar $\boldsymbol{R}$(T, L) a FNBC teniendo en cuenta que: $\boldsymbol{T}$ = {A, B, C, D}, $\boldsymbol{L}$ = {A ➞ C, B ➞ D}
  $\boldsymbol{K}$ = {AB}



La descomposición ρ={R_1, R_2, R_3} garantiza que los tres esquemas están en FNBC

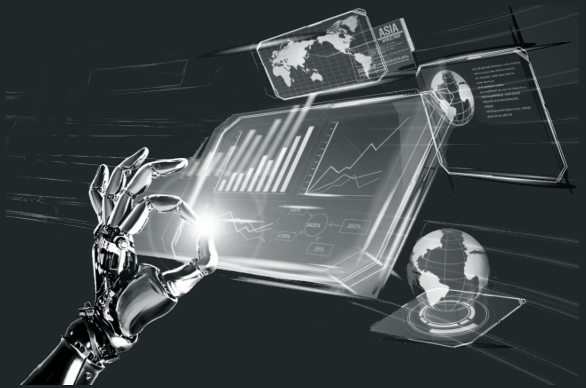
Bibliografía

Ramez A. Elmasri, Shamkant B. Navathe. Fundamentos de Sistemas de Bases de Datos (5º edic.). Prentice-Hall. 2007, [Cap. 10]

de Miguel, A.; Piattini, M. Fundamentos y modelos de bases de datos (2ª edic. Ra-Ma, 1999, [Cap. 8]

A. Silberschatz, Korth, Sudarshan. Fundamentos de Bases de Datos (5ª edic.). McGraw-Hill [cap. 7]

Ruiz, Francisco. Teoría de la Normalización [cap. 7] www.inf-cr.uclm.es/www/fruiz/bda/doc/teo/bda-t71.pdf



Gestión de Datos para Robótica

T1e - Normalización

Álvaro Vázquez Álvarez
Departamento de Electrónica e Computación

alvaro.vazquez@usc.es

Pabellón III - Despacho 4

Curso 2023-2024