DISEÑO DE BASES DE DATOS RELACIONALES:

MODELADO LÓGICO: EL MODELO RELACIONAL

Tema 3

Jorge García Duque

Índice

- Modelo Relacional
- Transformación del Modelo E-A al Modelo Relacional.
- Álgebra relacional
- Normalización

Modelo Relacional (I)

- Fue introducido por Codd en 1970. Todo el modelo tiene un fuerte apartado matemático subyacente.
- Una Base de Datos en el modelo relacional es un conjunto de relaciones.
- Relación:
  - información relativa a un conjunto de elementos homogéneos.
  - representada en filas y columnas.
    - Las filas las consideraremos registros o tuplas.
    - Las columnas las consideraremos campos o atributos.
    - Siempre debe haber una clave para identificar de forma única a los registros.

Modelo Relacional (II)

- Formalmente, una **relación *r*** es un un conjunto de m-tuplas (a_1, a_2, …, a_n) donde cada $a_i \in D_i$ (siendo cada D_i un conjunto). Es decir, una **relación *r*** es un subconjunto de D_1 x D_2 x … x D_n
- Los valores que almacene en cada momento (**instancia de la relación**) se especifican en forma tabular.
  - Un elemento *t* de *r* es una **tupla** y se representa por una fila.
  - El orden de las tuplas no es relevante (las tuplas se pueden almacenar en un orden arbitrario)
- Un **esquema de relación** identifica, mediante un nombre único para cada **atributo**, el conjunto de atributos de dicha *relación:* R = (A_1, A_2, …, A_n).
  - P.e. *Docente* = (*ID, nombre, nombre_dpto, salario*)
- El conjunto de valores permitidos para cada atributo se denomina **dominio** del atributo.
  - El valor *null (desconocido/sin valor)* pertenece a todos los dominios.
- Los valores de los atributos deben ser **atómicos**, es decir, indivisibles.
  - Atributos multivalorados o compuestos no son atómicos.
- Claves:
  - Como en el modelo E-A*: **superclave, clave candidata y clave primaria (PK).***
  - ***Clave Foránea (FK):*** restricción del valor de un atributo en una relación que tiene que existir en otra.

ID	*nombre*	*nombre_dpto*	*salario*
22222	Ana	Física	45000
12121	Felipe	Economía	40000
32343	Juan	Historia	30000
45565	María	Telemática	35000
98345	José	Señal y Comun.	40000
76766	Asunción	Biología	32000
10101	Mariano	Telemática	35000
58583	Loreto	Historia	50000
83821	Pedro	Telemática	47000
15151	Luis	Música	33000
33456	Paula	Física	40000
76543	Lucía	Economía	42000

(a) Relación *profesores*

Transformación del Modelo E-A al Modelo Relacional (I): Atributos no atómicos

- Los atributos compuestos se eliminan creando un nuevo atributo para cada uno de los campos componentes
  - P.e. dado el conjunto entidad *docente* con atributo compuesto *nombre* con atributos componentes *nombre_comun* y *primer_apellido, el esquema de relación* correspondiente al conjunto entidad tendrá dos atributos
    nombre_nombre_comun y *nombre_primer_apellido*
    - *El prefijo se puede omitir si no hay ambiguedad*
- Un atributo multivalorado M de un conjunto entidad E se representa mediante un nuevo esquema de relación EM
  - EM tendrá como atributos la clave primaria de E y un atributo que se corresponderá con el atributo multivalorado M.
  - El conjunto Entidad E se representa por un esquema de relación sin el atributo multivalorado.
  - P.e. El atributo multivalorado *telefonos* de *Docente* se representa mediante el esquema de relación *Docente_telefonos* (<u>*ID*</u>, <u>*numero_telefono*</u>)
  - Cada valor de un atributo multivalorado se corresponde con una fila diferente de la relación EM
    - P.e., una entidad docente con clave primaria “22222” y teléfonos “123456” y “234567” se corresponde con dos tuplas en la nueva relación:
      (22222, 123456) y(22222, 234567)

Transformación del Modelo E-A al Modelo Relacional (II): Conjuntos Entidad Fuertes y Débiles

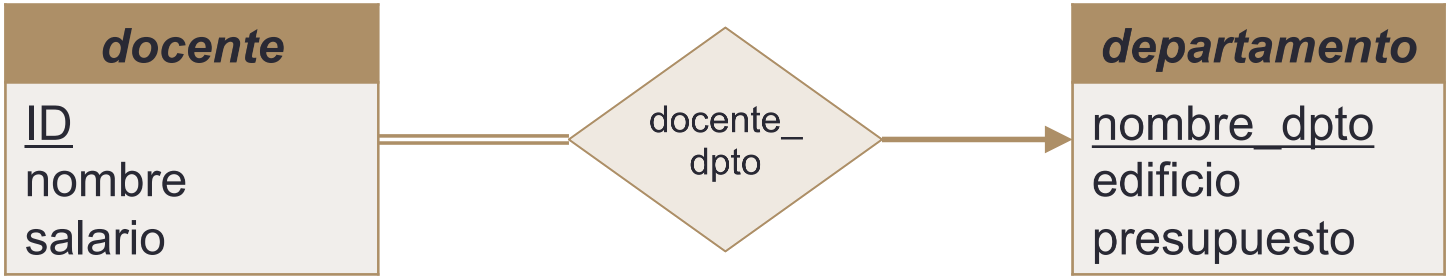
- Un conjunto entidad fuerte se transforma en un esquema de relación con los mismos atributos.
  - La clave primaria del conjunto entidad pasa a ser la clave primaria del esquema de relación.
  - P.e.: *Alumno(ID, nombre, tot_creditos)*
- Un conjunto entidad débil se convierte en un esquema de relación que incluye una columna para la clave primaria del conjunto entidad fuerte que la identifica.
  - La clave primaria del esquema de relación está formada por la unión de la clave primaria del conjunto entidad fuerte y el discriminador del conjunto entidad débil.
  - P.e. *Grupo(id_materia, id_grupo, cuatrimestre, año)*

Transformación del Modelo E-A al Modelo Relacional (III): Conjuntos Asociación varios a varios

- Un conjunto asociación varios a varios se representa con un esquema de relación con atributos para las claves primarias de los dos conjuntos entidad participantes (y también con los atributos descriptivos del conjunto asociación si los hubiera).
- La unión de las claves primarias de los conjuntos entidad participantes en la asociación pasa a ser la clave primaria del esquema de relación, y cada una de ellas es una clave foránea referenciando la clave primaria correspondiente en cada conjunto entidad participante.
- P.e.: *Tutoriza(id_a, id_d)*

Transformación del Modelo E-A al Modelo Relacional (IV): Conjuntos Asociación uno a varios y uno a uno

- Un conjunto asociación uno a varios se representa añadiendo un atributo adicional en el esquema de relación correspondiente al conjunto entidad del lado de "varios". Dicho atributo debe ser clave foránea y referenciar la clave primaria del esquema de relación correspondiente al lado de "uno".
  - Si la participación es parcial en el lado de "varios", puede dar lugar a la aparición de valores *null.*
  - P.e.: añadimos una clave foránea *nombre_dpto* al esquema de relación correspondiente al conjunto entidad *docente*.
- Para un conjunto asociación uno a uno cualquiera de los extremos puede jugar el papel de "varios".
  - Es decir, el atributo adicional (clave foránea) se puede añadir a cualquiera de los esquemas correspondientes a los dos conjuntos entidad
- En el esquema correspondiente a un conjunto asociación, el enlace entre la entidad débil y su entidad fuerte identificadora es redundante.
  - P.e. El esquema de relación *grupo* ya contiene los atributos que deberían aparecer en el posible esquema de relación Grupo*_materia*.



Transformación del Modelo E-A al Modelo Relacional (V): Especialización/Generalización

- La clave primaria del conjunto entidad generalizada se incluye como clave primaria en los esquemas de relación correspondientes a los conjuntos entidad especializados.
- En los esquemas de relación especializados se incluyen restricciones de FK a la PK del esquema de relación correspondiente al conjunto entidad generalizado.
- Opciones:
  - Método 1: Cada esquema de relación correspondiente a un conjunto entidad especializado incluye solo los atributos especializados:

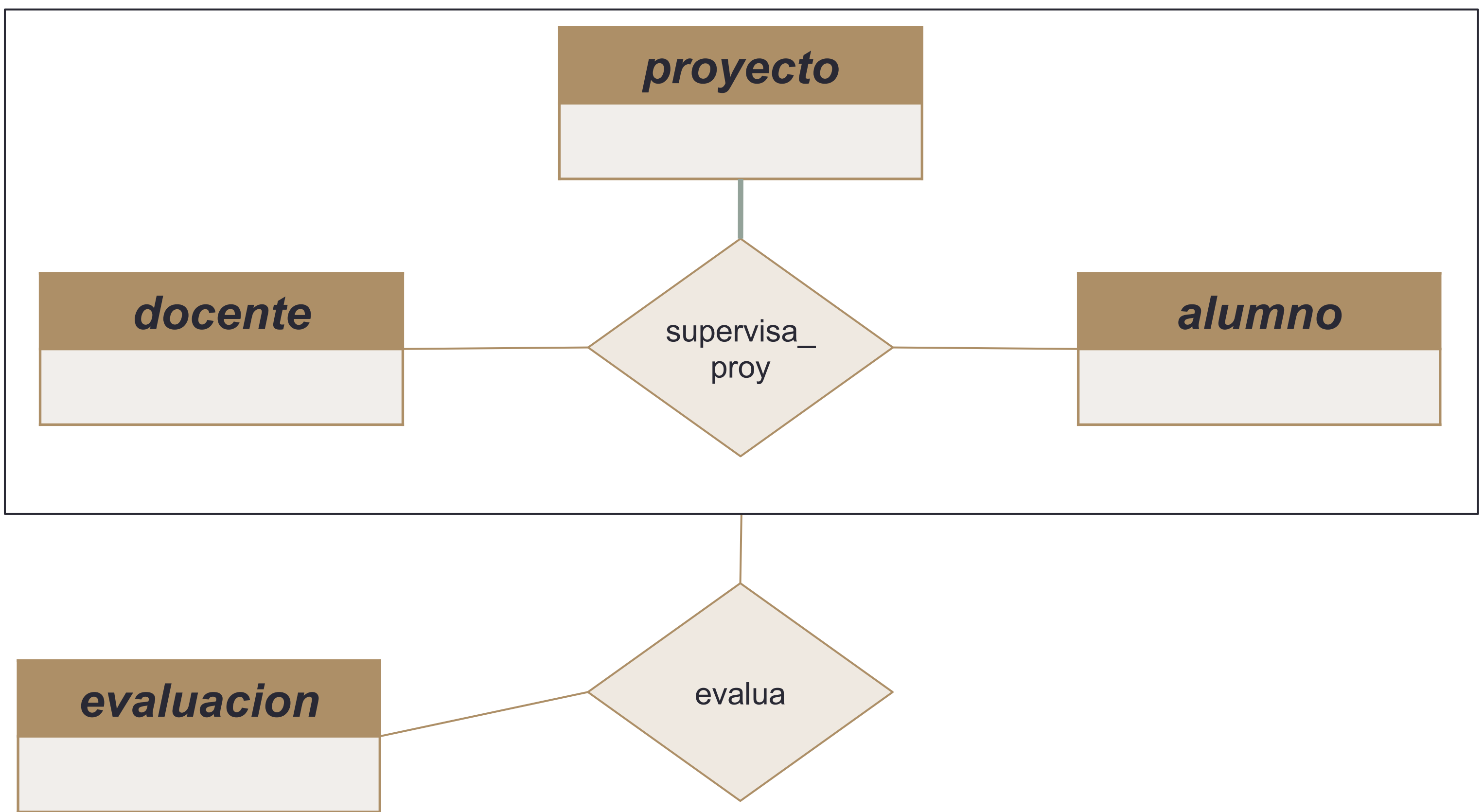
esquema	atributos
Persona	*ID, nombre, calle, ciudad*
Alumno	*ID, tot_creditos*
Empleado	*ID, salario*

  - Método 2 (si la especialización es disjunta y completa): Cada esquema de relación correspondiente a un conjunto entidad especializado incluye todos los atributos heredados
    - El esquema de relación correspondiente al conjunto entidad generalizado es necesario solo para incluir las restricciones de FK:

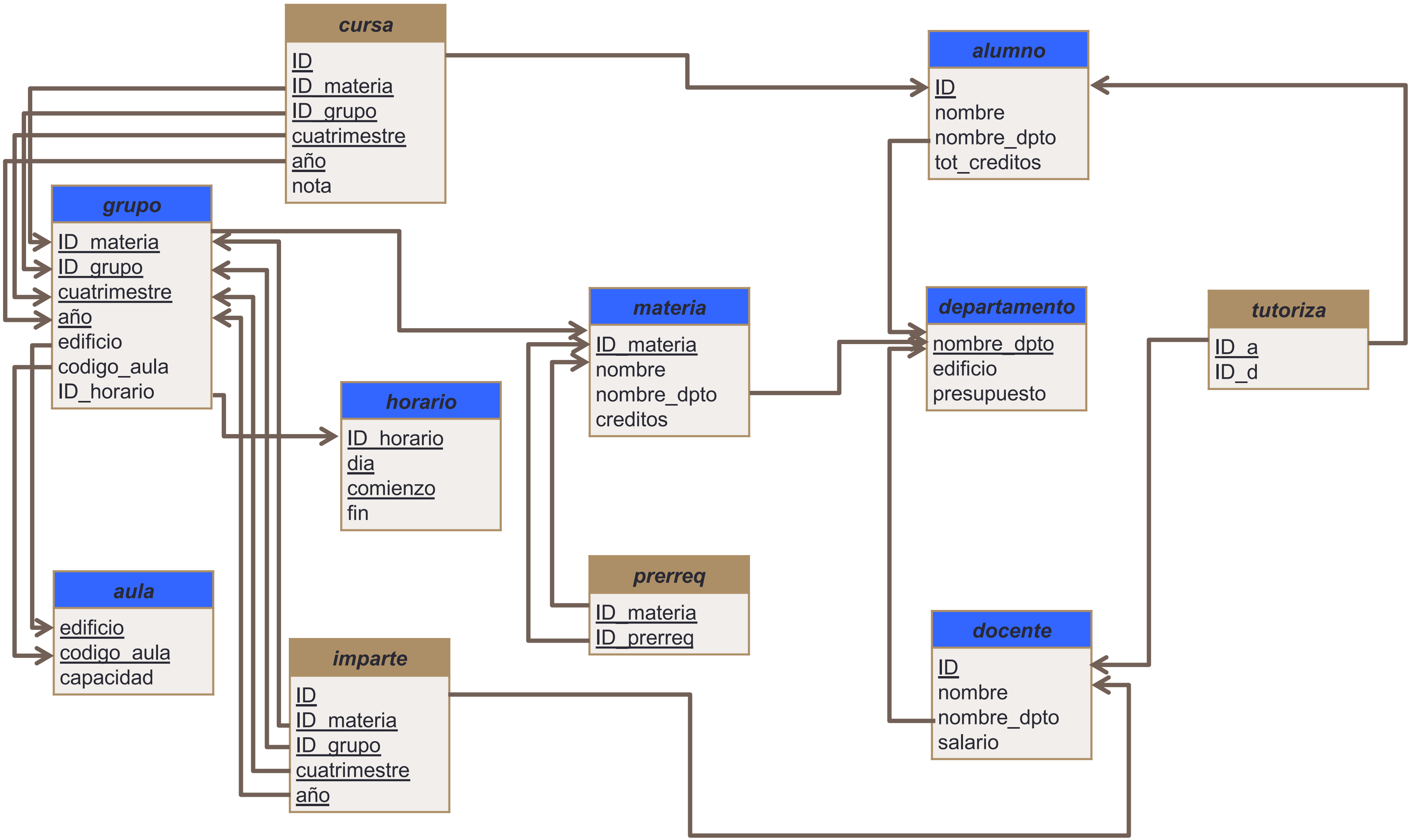
esquema	atributos
Persona	*ID*
Alumno	*ID, nombre, calle, ciudad, tot_creditos*
Empleado	*ID, nombre, calle, ciudad, salario*

Transformación del Modelo E-A al Modelo Relacional (VI): Agregación

- Para representar una agregación, se crea un esquema de relación con:
  - Clave primaria: la clave primaria de la asociación agregada más la clave primaria del conjunto entidad asociado.
  - Los atributos del conjunto entidad asociado.
  - P.e.: *Evalua (ID_a, ID_proyecto, ID_d, ID_evaluacion)*



Transformación del Modelo E-A al Modelo Relacional (VII): Diagrama del Esquema Lógico de la Base de Datos



Transformación del Modelo E-A al Modelo Relacional (VIII): Restricciones de Integridad

- Las restricciones de integridad nos protegen ante daños accidentales en la base de datos, asegurando que los cambios autorizados en la base de datos no van a producir una pérdida de consistencia en los datos
  - restricciones de dominios: comprueban los valores insertados en la base de datos *(check),* y comprueban las consultas para asegurar que las comparaciones (*tipos)* tienen sentido.
  - Integridad Referencial: asegura que un valor que aparece en una relación para un conjunto de atributos determinado también aparece en un conjunto de atributos de otra relación
    - Dadas las relaciones $r_1(R_1)$ y $r_2(R_2)$ con claves primarias K_1 y K_2 respectivamente.
    - El subconjunto α de R_2 es una ***clave foránea*** referenciando K_1 en la relación r_1, si para cada t_2 en r_2 debe haber una tupla t_1 en r_1 tal que $t_1[K_1] = t_2[\alpha]$.
    - Las restricciones de integridad referencial también se denominan dependencias de subconjunto ya que se pueden expresar como

$$\prod_{\alpha} (r_2) \subseteq \prod_{K1} (r_1)$$

Transformación del Modelo E-A al Modelo Relacional (IX): Comprobación de Integridad Referencial

- **Insertar.** Si una tupla t_2 se inserta en r_2, el sistema se debe asegurar de que hay una tupla t_1 en r_1 tal que $t_1[K_1] = t_2[\alpha]$. Es decir

$$t_2\ [\alpha] \in \prod_K (r_1)$$

- **Eliminar.** Si se elimina una tupla t_1 de r_1, el sistema debe hallar el conjunto de tuplas de r_2 referenciando t_1:

$$\sigma_{\alpha\ =\ t1[K]}\ (r_2)$$

  Si el conjunto no es vacío
  - o bien se rechaza el comando como un error,
  - o bien se deben eliminar las tuplas que referencian a t_1 (se permiten *eliminaciones en cascada*)

- **Actualizaciones.** Hay dos casos:
  - Si se actualiza una tupla t_2 en la relación r_2 y la actualización modifica los valores de la clave foránea α,entonces se debe hacer un test similar al caso de inserción:
    - Si t_2' denota el nuevo valor de la tupla t_2, el sistema se debe asegurar de que

$$t_2'[\alpha] \in \prod_K(r_1)$$

  - Si se actualiza una tupla t_1 en r_1, y la actualización modifica el valor de la clave primaria (K), entonces se debe realizar un test similar al del caso de eliminación utilizando el valor anterior de t1 (el valor antes de hacer la actualización).

$$\sigma_{\alpha\ =\ t1[K]}\ (r_2)$$

    Si el conjunto no es vacío
    - la actualización se puede rechazar como un error, o
    - la actualización se puede hacer en cascada sobre las tuplas del conjunto, o
    - se eliminan las tuplas del conjunto.

Álgebra Relacional

- Lenguajes formales de consulta relacional:
  - Álgebra relacional (procedimental), consiste en un conjunto de operaciones sobre relaciones (instancias), donde el resultado es otra relación (instancia), siendo la base del lenguaje SQL:
    - Operaciones Básicas
      - Selección
      - Proyección
      - Unión
      - Diferencia de conjuntos
      - Producto cartesiano
      - Renombrar
    - Operaciones Añadidas (no aportan expresividad sobre las básicas)
      - Intersección
      - *Join* natural
      - Asignación
      - División
    - Operaciones Extendidas
      - Proyección Generalizada
      - Agregación
      - *Join* externo (outer)
  - Cálculo relacional (no procedimental), basado en el cálculo de predicados de primer orden.
    - Cálculo relacional de tuplas
    - Cálculo relacional de dominios

Álgebra Relacional: Selección(I)

- Notación: $\sigma_p(r) = \{t \mid t \in r \ \mathbf{y}\ p(t)\}$
- El predicado de la selección *p* es una fórmula en calculo proposicional formada por términos unidos por: ∧ (y), ∨ (**o**), ¬ (**no**)
- Cada término tiene la forma:

  <atributo> *op* <atributo> o <constante>

  donde *op* es: =, ≠, >, ≥, <, ≤
- P.e. Seleccionar los departamentos de un nombre:

  $\sigma_{nombre\_dpto=\text{“Ingeniería Telemática”}}(departamento)$

Álgebra Relacional: Selección (II)

- Relación *r*

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

- $\sigma_{A=B \wedge D > 5}(r)$

A	B	C	D
α	α	1	7
β	β	23	10

Álgebra Relacional: Proyección(I)

- Notación: $\prod_{A1,\ A2,\ \ldots,\ Ak} (r)$,
- $A_1 \ldots A_k$ son atributos de *r*.
- El resultado es la relación de *k* columnas que se obtiene eliminando las columnas no listadas
- Las filas duplicadas del resultado se eliminan (el resultado debe ser una relación)
- P.e. Eliminar el atributo nombre-sucursal de *cuentas*
  $\prod_{numero\text{-}cuenta,\ saldo}$ (cuenta)

Álgebra Relacional: Proyección(II)

- Relación *r*:

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

$\prod_{A,C}(r)$

A	C
α	1
α	1
β	1
β	2

=

A	C
α	1
β	1
β	2

Álgebra Relacional: Unión(I)

- Notación: $r \cup s = \{t \mid t \in r \text{ o } t \in s\}$
- Para que $r \cup s$ sea válida, ambas relaciones deben ser *compatibles:*
  - *r, s* deben tener el mismo número de atributos.
  - Los dominios de los atributos deben ser *compatibles* (p.e., la 2ª columna de r contiene el mismo tipo de valores que la 2ª columna de *s*)
- P.e. encontrar los nombres de todas las personas de la universidad

$$\prod_{nombre}(alumno) \cup \prod_{nombre}(docente)$$

Álgebra Relacional: Unión(II)

- Relaciones *r, s:*

A	*B*
α	*1*
α	*2*
β	*1*

r

A	*B*
α	*2*
β	*3*

s

r **U** s:

A	*B*
α	*1*
α	*2*
β	*1*
β	*3*

Álgebra Relacional: Diferencia(I)

- Notación: $r - s = \{t \mid t \in r$ **y** $t \notin s\}$
- La diferencia de conjuntos se debe realizar entre relaciones *compatibles*.
  - *r* y *s* deben tener el mismo número de atributos
  - Los dominios de los atributos de *r y s* deben ser compatibles

Álgebra Relacional: Diferencia(II)

- Relaciones *r, s:*

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

$r - s$:

A	B
α	1
β	1

Álgebra Relacional: Renombrado

- Permite nombrar, y por tanto referirnos a, los resultados de las expresiones de álgebra relacional.
  - Permite referirse a una relación con más de un nombre.
- Ejemplo:

$$\rho_X (E)$$

devuelve la expresión *E* con el nombre *X*

- Si la expresión *E* en álgebra relacional tiene un orden *n*, entonces

$$\rho_{X\ (A1,\ A2,\ \ldots,\ An)} (E)$$

devuelve la expresión *E* con el nombre *X*, y con los atributos renombrados a A_1, *A2*, ….., *An*.

Álgebra Relacional: Producto Cartesiano(I)

- Notación: $r \times s = \{(t,q) \mid t \in r$ **y** $q \in s\}$
- Dado que el resultado debe ser una relación, los atributos de r(R) y s(S) deben ser disjuntos: $R \cap S = \emptyset$.
    - Si los atributos de *r(R)* y *s(S)* no son disjuntos, se debe utilizar la operación de renombrar.

Álgebra Relacional: Producto Cartesiano(II)

Relaciones *r*, *s*:

A	B
α	1
β	2

r

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

s

r x *s*:

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

Álgebra Relacional: Combinación de operaciones

- Se pueden construir expresiones utilizando varias operaciones
- Ejemplo: $\sigma_{A=C}(r\ x\ s)$
- $r\ x\ s$

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

- $\sigma_{A=C}(r\ x\ s)$

A	B	C	D	E
α	1	α	10	a
β	2	β	10	a
β	2	β	20	b

Álgebra Relacional: Resumen y Definición Formal

- Una expresión básica en álgebra relacional puede ser:
  - Una relación de la base de datos
  - Una relación constante
- Dadas dos expresiones en álgebra relacional E_1 y E_2 también son expresiones en álgebra relacional:
  - $E_1 \cup E_2$
  - $E_1 - E_2$
  - $E_1 \times E_2$
  - $\sigma_p (E_1)$, P es un predicado sobre los atributos de E_1
  - $\prod_S(E_1)$, S es una lista que contiene algunos atributos de E_1
  - $\rho_x (E_1)$, x es el nuevo nombre del resultado de E_1

Álgebra Relacional: Intersección(I)

- Notación: $r \cap s = \{ t \mid t \in r$ **and** $t \in s \}$
- La intersección de conjuntos se debe realizar entre relaciones *compatibles*.
  - *r* y *s* deben tener el mismo número de atributos
  - Los dominios de los atributos de *r y s* deben ser compatibles
- Se puede expresar mediante operadores básicos:

$$r \cap s = r - (r - s)$$

Álgebra Relacional: Intersección(II)

- Relaciones r, s:

r

A	B
α	1
α	2
β	1

s

A	B
α	2
β	3

- $r \cap s$

A	B
α	2

Álgebra Relacional: *Join* Natural(I)

- Notación: $r \bowtie s$

- Dadas dos relaciones *r* y *s* sobre los esquemas *R* y *S* respectivamente.
  Entonces, $r \bowtie s$ es una relación sobre el esquema $R \cup S$ que se obtiene de la siguiente manera:
  - Se considera cada par de tuplas t_r de *r* y t_s de *s*.
  - Si t_r y t_s tienen el mismo valor para cada uno de los atributos en $R \cap S$, se añade una tupla *t* al resultado, donde
    - *t* tiene el mismo valor que t_r sobre *r*
    - *t* tiene el mismo valor que t_s sobre *s*

- Ejemplo:
  $R = (A, B, C, D)$
  $S = (E, B, D)$
  - Esquema de relación = (A, B, C, D, E)
  - $r \bowtie s$ se define como:

$$\Pi_{r.A,\ r.B,\ r.C,\ r.D,\ s.E}\ (\sigma_{r.B\ =\ s.B\ \wedge\ r.D\ =\ s.D}\ (r \times s))$$

Álgebra Relacional: *Join* Natural(II)

- Relaciones r, s:

A	B	C	D
α	1	α	a
β	2	γ	a
γ	4	β	b
α	1	γ	a
δ	2	β	b

r

B	D	E
1	a	α
3	a	β
1	a	γ
2	b	δ
3	b	ε

s

$r \bowtie s$

A	B	C	D	E
α	1	α	a	α
α	1	α	a	γ
α	1	γ	a	α
α	1	γ	a	γ
δ	2	β	b	δ

Álgebra Relacional: División(I)

$$r \div s$$

- Adecuada para consultas que incluyan la expresión “para todos”.
- Dadas las relaciones *r* y *s* sobre los esquemas R y S respectivamente, donde
  - $R = (A_1, \ldots, A_m, B_1, \ldots, B_n)$
  - $S = (B_1, \ldots, B_n)$

  El resultado de r ÷ s es una relación sobre el esquema $R - S = (A_1, \ldots, A_m)$

$$r \div s = \{t \mid t \in \prod_{R-S}(r) \land \forall u \in s \ (t,u) \in r\}$$

Álgebra Relacional: División(II)

Relaciones *r*, *s*:

A	B
α	1
α	2
α	3
β	1
γ	1
δ	1
δ	3
δ	4
ε	6
ε	1
β	2

r

B
1
2

s

$r \div s$:

A
α
β

Álgebra Relacional: División(III)

Relaciones *r*, *s*:

A	*B*	*C*	*D*	*E*
α	a	α	a	*1*
α	a	γ	a	*1*
α	a	γ	b	*1*
β	a	γ	a	*1*
β	a	γ	b	*3*
γ	a	γ	a	*1*
γ	a	γ	b	*1*
γ	a	β	b	*1*

r

D	*E*
a	*1*
b	*1*

s

r ÷ *s*:

A	*B*	*C*
α	a	γ
γ	a	γ

Álgebra Relacional: Asignación

- La operación asignación ($\leftarrow$) facilita un modo útil para expresar consultas complejas:
  - Escribir consultas como un programa secuencial consistente en
    - un conjunto de asignaciones
    - seguido por una expresión cuyo valor se muestre como el resultado de la consulta.
  - La asignación siempre se debe realizar a una variable relación temporal.
- Ejemplo: $r \div s$ se puede expresar como

$$
\begin{aligned}
&temp1 \leftarrow \Pi_{R-S}(r) \\
&temp2 \leftarrow \Pi_{R-S}((temp1 \times s) - \Pi_{R-S,S}(r)) \\
&result = temp1 - temp2
\end{aligned}
$$

  - El resultado de la expresión a la derecha de $\leftarrow$ se asigna a la variable relación de la izquierda de $\leftarrow$.
  - La variable se puede utilizar en las expresiones que van a continuación.

Álgebra Relacional: Operaciones Extendidas. Proyección Generalizada

- Extiende la operación de proyección permitiendo funciones aritméticas en la lista de proyección.

$$\Pi_{F1,\ F2,\ \ldots,\ Fn}(E)$$

- *E* es cualquier expresión en álgebra relacional
- $F_1, F_2, \ldots, F_n$ son expresiones aritméticas que incluyen constantes y atributos del esquema de *E*.
- P.e. Dada la relación *info_producto(nombre, precio_compra, precio_venta),* averiguar cuánto se gana en cada producto:

$$\Pi_{nombre,\ precio\_venta\ -\ precio\_compra}\ (info\_producto)$$

Álgebra Relacional: Operaciones Extendidas. Agregación(I)

- Utiliza **funciones de agregación** que toman como argumentos un conjunto de valores y devuelven un valor simple como resultado.

  avg: valor medio
  min: valor mínimo
  max: valor máximo
  sum: suma de valores
  count: número de valores

- **Operación agregada** en álgebra relacional

$$_{G1,\, G2,\, \ldots,\, Gn}\, \mathcal{G}\, _{F1(A1),\, F2(A2),\ldots,\, Fn(An)}\, (E)$$

  - E es cualquier expresión en álgebra relacional
  - G_1, G_2 …, G_n es una lista de atributos sobre los que agrupar (puede estar vacía)
  - Cada F_i es una función agregada
  - Cada A_i es un nombre de atributo

Álgebra Relacional: Operaciones Extendidas. Agregación(II)

- Relación *r*:

A	*B*	*C*
α	α	7
α	β	7
β	β	3
β	β	10

$$\mathcal{G}_{\mathbf{sum(c)}}\,(\mathrm{r})$$

sum(c)
27

Álgebra Relacional: Operaciones Extendidas. Agregación(III)

- Relación *cuenta* agrupada por *nombre-sucursal*:

nombre-sucursal	*numero-cuenta*	*saldo*
Vigo	A-102	400
Vigo	A-201	900
Madrid	A-217	750
Madrid	A-215	750
Pontevedra	A-222	700

$$_{nombre\text{-}sucursal}\,\mathcal{G}_{\,sum(saldo)}\,(cuenta)$$

nombre-sucursal	*sum(saldo)*
Vigo	1300
Madrid	1500
Pontevedra	700

Álgebra Relacional: Operaciones Extendidas. Agregación(IV)

- El resultado de la agregación no tiene nombre
  - Se puede utilizar la operación de renombrado para darle un nombre
  - Por conveniencia, se permite el renombrado como parte de la operación de agregación

$$_{\textit{nombre-sucursal}}\,\mathcal{G}\,_{\textbf{sum}(\textit{saldo})\ \textbf{as}\ \textit{sum-saldo}}(cuenta)$$

Álgebra Relacional: Operaciones Extendidas. *Join* Externo *(outer)* (I)

- Es una extensión de la operación de join que evita la pérdida de información.
- Calcula el join y después añade las tuplas de una relación que no coinciden con las tuplas de la otra relación al resultado del join.
- Utiliza valores *null*:
  - *null* significa que el valor es desconocido o no existe
  - Todas la comparaciones en las que participa un valor *null* son **falsas** por definición.

Álgebra Relacional: Operaciones Extendidas. *Join* Externo *(outer)* (II)

- Relación *materia*

ID_materia	*nombre_materia*	*creditos*
170	Biología	3
230	Geografía	4
260	Matemáticas	6

- Relación *matriculado*

nombre_alumno	*ID_materia*
Juan López	L-170
Ana Vázquez	L-230
José García	L-155

Álgebra Relacional: Operaciones Extendidas. *Join* Externo *(outer)* (III)

- **Join interno**

materia ⋈ *matriculado*

ID_materia	*nombre_materia*	*creditos*	*nombre_alumno*
170	Biología	3	Juan López
230	Geografía	4	Ana Vázquez

- **Join externo izquierdo**

materia ⟕ *matriculado*

ID_materia	*nombre_materia*	*creditos*	*nombre_alumno*
170	Biología	3	Juan López
230	Geografía	4	Ana Vázquez
260	Matemáticas	6	*null*

Álgebra Relacional: Operaciones Extendidas. *Join* Externo *(outer)* (IV)

- **Join externo derecho**

materia ⋊ *matriculado*

ID_materia	*nombre_materia*	*creditos*	*nombre_alumno*
170	Biología	3	Juan López
230	Geografía	4	Ana Vázquez
155	*null*	*null*	José García

- **Join externo total**

materia ⟗ *matriculado*

ID_materia	*nombre_materia*	*creditos*	*nombre_alumno*
170	Biología	3	Juan López
230	Geografía	4	Ana Vázquez
260	Matemáticas	6	*null*
155	*null*	*null*	José García

Álgebra Relacional: Valores *null (I)*

- Las tuplas pueden contener valores nulos, denotados por *null*, en algunos de sus atributos
- *null* significa valor desconocido o que el valor no existe.
- El resultado de cualquier expresión aritmética en la que participe *null* es *null.*
- Las funciones agregadas ignoran los valores *null*
  - Es una decisión arbitraria. Alternativamente se podría haber devuelto como resultado *null*.
  - Seguimos la semántica de SQL respecto al manejo de valores nulos
- Para eliminación de duplicados y agrupamientos, *null* recibe el mismo tratamiento que cualquier otro valor

Álgebra Relacional: Valores *null (II)*

- Las comparaciones con valores *null* devuelven un valor especial de verdad denominado *desconocido*
  - Si se usa *falso* en vez de *desconocido*, entonces *not (A < 5)* no sería equivalente a *A >= 5*
- Lógica trivalorada utilizando el valor de verdad *desconocido*:
  - OR: (*desconocido* **or** verdad) = verdad,
    (*desconocido* **or** *falso*) = *desconocido*,
    (*desconocido* **or** *desconocido)* = *desconocido*
  - AND: *(verdad* **and** *desconocido)* = *desconocido*,
    (falso **and** *desconocido)* = *falso*,
    (desconocido **and** *desconocido)* = *desconocido*
  - NOT*:* (**not** *desconocido)* = *desconocido*
  - En SQL "*P* **es desconocido**" se evalúa a verdad si el predicado *P* se evalúa a *desconocido*
- El resultado de un predicado de selección se trata como *falso si se evalúa como desconocido*

Álgebra Relacional: Modificación de la base de datos

- El contenido de la base de datos se puede modificar utilizando las siguientes operaciones:
  - Borrado
  - Inserción
  - Actualización
- Todas estas operaciones se expresan mediante el operador de asignación.

Álgebra Relacional: Borrado(I)

- Una petición de borrado se expresa de manera similar a una consulta, excepto que, en vez de mostrar las tuplas al usuario, las tuplas seleccionadas se eliminan de la base de datos.
- Solo se pueden eliminar tuplas completas; no se pueden eliminar solo determinados atributos
- Un borrado se expresa en álgebra relacional como:

$$r \leftarrow r - E$$

donde r es una relación y E es una consulta en álgebra relacional.

Álgebra Relacional: Borrado(II)

- Borrar todas los alumnos con menos de 50 créditos cursados.

$$alumno \leftarrow alumno - \sigma_{tot\_creditos < 50}(alumno)$$

Álgebra Relacional: Inserción(I)

- Para insertar datos en una relación podemos:
  - o bien especificar la tupla a insertar
  - o bien escribir una consulta cuyo resultado esté formado por las tuplas a insertar
- En álgebra relacional, una inserción se expresa:

$$r \leftarrow r \cup E$$

  donde r es una relación y E es una expresión en álgebra relacional.

- La inserción de una sola tupla se realiza cuando E es una relación constante que contiene una tupla.

Álgebra Relacional: Inserción(II)

- Insertar información en la base de datos sobre un nuevo alumno.

$$alumno \leftarrow alumno \cup \{(\text{“76767”}, \text{“Felipe Gómez”}, \text{“Ingeniería Telemática”}, 0)\}$$

Álgebra Relacional: Actualización(I)

- Permite cambiar el valor de una tupla sin cambiar *todos* los valores de la tupla
- Para ello se utiliza la operación de proyección generalizada

$$r \leftarrow \Pi_{F1,\, F2,\, \ldots,\, Fl,} (r)$$

- Cada F_i es
  - el atributo *i* de *r*, si el atributo *i* no se quiere actualizar, o,
  - si se va a actualizar el atributo *i*, F_i es una expresión en la que intervienen solamente constantes y los atributos de *r*, que proporciona el nuevo valor del atributo

Álgebra Relacional: Actualización(II)

- Añadir 5 créditos al alumno con ID “22222”.

$$alumno \leftarrow \Pi_{ID,\ nombre,\ nombre\_dpto,\ tot\_creditos\ +\ 5}(\sigma_{ID=\text{“}22222\text{”}}(alumno))$$

Normalización: Introducción

- Metodología para decidir cuándo una determinada relación *R* presenta una "*forma adecuada*", evitando redundancia o problemas en las actualizaciones de la base de datos.
- Se definen diferentes *formas normales:*
  - 1FN ⇒ 2FN ⇒ 3FN ⇒ *FNBC* ⇒ 4FN ⇒ 5FN
- Para poder decidir si un esquema de relación R está en una forma normal utilizaremos las restricciones implícitas en el modelo relacional y nuevas restricciones: *dependencias funcionales.*
  - Indican que el valor de un cierto conjunto de atributos determina de manera unívoca el valor de otro conjunto de atributos (generalización del concepto de *clave*). **$\alpha \rightarrow \beta$**
- En el caso de que la relación *R* no esté en una "*forma adecuada*", se descompone en un conjunto de relaciones $\{R_1, R_2, \ldots, R_n\}$ tales que:
  - Cada relación presente una *"forma adecuada"*.
  - La descomposición sea sin pérdidas (reversible por join).
  - Se preserven las dependencias funcionales.

Normalización: Descomposición con pérdidas

ID	*nombre*	*calle*	*ciudad*	*salario*
…				
57766	Ana	Florida	Madrid	75000
98776	Ana	Norte	Barcelona	67000
…				

empleado

ID	*nombre*
…	
57766	Ana
98776	Ana
…	

nombre	*calle*	*ciudad*	*salario*
Ana	Florida	Madrid	75000
Ana	Norte	Barcelona	67000

Join natural

ID	*nombre*	*calle*	*ciudad*	*salario*
…				
57766	Ana	Florida	Madrid	75000
57766	Ana	Norte	Barcelona	67000
98776	Ana	Florida	Madrid	75000
98776	Ana	Norte	Barcelona	67000
…				

Normalización: 1FN y 2FN

- Un esquema de relación R está en **primera forma normal** (**1FN**) si los dominios de todos sus atributos son atómicos (implícito en el modelo relacional).
- Un esquema de relación R está en **segunda forma normal** (**2FN**) si está en 1FN y todos los atributos que no forman parte de una clave candidata dependen funcionalmente de una clave candidata completa (no existen dependencias parciales de claves candidatas).

ID	*nombre*	*salario*	*nombre_dpto*	*edificio*	*presupuesto*
22222	Ana	45000	Física	Fac. Ciencias	90000
12121	Felipe	40000	Economía	Fac. Económicas	120000
32343	Juan	30000	Historia	Fac. Humanidades	40000
45565	María	35000	Telemática	Esc. Teleco	230000
98345	José	40000	Señal y Comun.	Esc. Teleco	180000
76766	Asunción	32000	Biología	Fac. Ciencias	95000
10101	Mariano	35000	Telemática	Esc. Teleco	230000
58583	Loreto	50000	Historia	Fac. Humanidades	40000
83821	Pedro	47000	Telemática	Esc. Teleco	230000
15151	Luis	33000	Música	Fac. Humanidades	12000
33456	Paula	40000	Física	Fac. Ciencias	90000
76543	Lucía	42000	Economía	Fac. Económicas	120000

Relación *Docente_dpto*

Normalización: Dependencias funcionales (I)

- Dado un esquema de relación R, $\alpha \subseteq R$ *y* $\beta \subseteq R$, la **dependencia funcional $\alpha \rightarrow \beta$ se cumple en R** si toda instancia r(R) cumple:
  - si dos tuplas cualquieras t_1 y t_2 de *r* coinciden en los valores de los atributos α, entonces también coinciden en los valores de los atributos β. Es decir:

$$t_1[\alpha] = t_2[\alpha] \Rightarrow t_1[\beta] = t_2[\beta]$$

- K es una superclave para el esquema de relación R si y sólo si
  - $K \rightarrow R$
- K es una clave candidata de R si y sólo si:
  - $K \rightarrow R$, y
  - no existe $\alpha \subset K$, $\alpha \rightarrow R$
- Permiten expresar restricciones que no se pueden expresar con superclaves.
- Ejemplo
  - *Docente_depto* (*<u>ID</u>*, *nombre*, *salario*, *<u>nombre_dpto</u>*, *edificio*, *presupuesto*).

nombre_depto → *edificio, presupuesto*

Normalización: Dependencias funcionales (II)

- Una dependencia funcional es **trivial** si se satisface para todas las instancias posibles de una relación.
  - Ejemplo:
    - *ID, nombre* $\rightarrow$ *ID*
    - *nombre* $\rightarrow$ *nombre*
  - En general, $\alpha \rightarrow \beta$ es trivial si $\beta \subseteq \alpha$
- Dado un conjunto de dependencias funcionales *F*, hay otras dependencias funcionales que se pueden inferir lógicamente a partir de *F*.
  - Por ejemplo: Si $A \rightarrow B$ y $B \rightarrow C$, entonces $A \rightarrow C$
- El conjunto de **todas** las dependencias funcionales inferidas a partir de *F* forma el **cierre** de *F* (**F^+**).

Normalización: FNBC(I)

- Un esquema de relación *R* está en Forma Normal de Boyce Codd (FNBC) con respecto a un conjunto de dependencias funcionales *F* si para todas las dependencias funcionales de F^+ $\alpha \rightarrow \beta$ se cumple al menos una de las siguientes condiciones:
    - $\alpha \rightarrow \beta$ es trivial (es decir, $\beta \subseteq \alpha$)
    - α es una superclave de *R*

- Dado un esquema *R* y una dependencia no trivial $\alpha \rightarrow \beta$ que viola la FNBC, descomponemos *R* en:
    - $(\alpha \cup \beta)$
    - $(R - (\beta - \alpha))$

Normalización: FNBC(II)

- *nombre_depto* → *edificio, presupuesto* pero *nombre_depto* no es una superclave
- Descomposición para cumplir FNBC:
  - (α U β) = (*nombre_depto*, *edificio, presupuesto*)
  - (*R* - (β - α)) = (*ID*, *nombre*, *salario*, *nombre_depto*)

ID	*nombre*	*salario*	*nombre_dpto*	*edificio*	*presupuesto*
22222	Ana	45000	Física	Fac. Ciencias	90000
12121	Felipe	40000	Economía	Fac. Económicas	120000
32343	Juan	30000	Historia	Fac. Humanidades	40000
45565	María	35000	Telemática	Esc. Teleco	230000
98345	José	40000	Señal y Comun.	Esc. Teleco	180000
76766	Asunción	32000	Biología	Fac. Ciencias	95000
10101	Mariano	35000	Telemática	Esc. Teleco	230000
58583	Loreto	50000	Historia	Fac. Humanidades	40000
83821	Pedro	47000	Telemática	Esc. Teleco	230000
15151	Luis	33000	Música	Fac. Humanidades	12000
33456	Paula	40000	Física	Fac. Ciencias	90000
76543	Lucía	42000	Economía	Fac. Económicas	120000

Relación *Docente_dpto*

Normalización: FNBC(III)

- Las restricciones, incluyendo las dependencias funcionales, son costosas de comprobar en la práctica, a menos que afecten a una sola relación.
- Si es suficiente con comprobar sólo las dependencias funcionales de cada relación individual de una descomposición para asegurar que **todas** las dependencias funcionales se cumplen, entonces la descomposición **preserva las dependencias**.
- No siempre es posible conseguir cumplir la FNBC y preservar las dependencias.
  - Para estos casos consideramos una forma normal más débil, denominada **tercera forma normal**.

Normalización: 3FN

- Un esquema de relación *R* está en **tercera forma normal (3FN)** con respecto a un conjunto de dependencias funcionales *F* si, para todas las $\alpha \rightarrow \beta$ de F^+ se cumple al menos una de las siguientes condiciones:
  - $\alpha \rightarrow \beta$ es trivial (es decir, $\beta \subseteq \alpha$)
  - α es una superclave de *R*
  - Cada atributo *A* de $\beta - \alpha$ forma parte de una clave candidata de *R.*

    (**NOTA**: cada atributo puede formar parte de una clave candidata distinta)
- Si una relación está en FNBC entonces está en 3FN (dado que en la FNBC se deben cumplir alguna de las dos primeras condiciones anteriores).
- La tercera condición representa la relajación mínima que es necesario realizar para asegurar que se preservan las dependencias.

Normalización: FNBC vs 3FN

- Siempre es posible descomponer un esquema en un conjunto de esquemas que estén en 3FN, de tal forma que:
  - la descomposición sea reversible por join.
  - se preservan las dependencias.
- Siempre es posible descomponer un esquema en un conjunto de esquemas que estén en FNBC, de tal forma que:
  - la descomposición sea reversible por join.
  - puede que no sea posible preservar las dependencias.
- Los objetivos del diseño de una base de datos relacional son:
  - FNBC.
  - Descomposición reversible por join.
  - Preservar dependencias.
- Si no se puede conseguir, debemos aceptar:
  - perder la preservación de dependencias, o
  - admitir redundancia y utilizar la 3FN.

Normalización: Otras formas normales de orden superior

- Las **dependencias multivaluadas** generalizan las dependencias funcionales.
  - dan lugar a la **cuarta forma normal** (**4FN**)
- Las **dependencias de join** generalizan las dependencias multivaluadas.
  - dan lugar a la **forma normal por proyección de join** (**FNPJ**) (también llamada **quinta forma normal 5FN**)
- Un tipo de restricciones aún más generales, dan lugar a una forma normal denominada **forma normal clave-dominio**.
- Problema de esta restricciones generalizadas: es difícil razonar sobre ellas y no existe un conjunto de reglas de inferencia consistente y completo.
- Muy raramente utilizadas.

FIN DEL TEMA 3