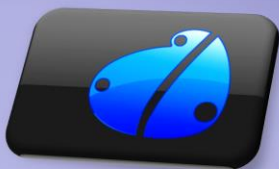


modelo relacional  
Tema 3



## el autor



*Edgar Frank Codd*

*1923 - 2003*

*1970, "A Relational Model of Data for Large  
Shared Data Banks"*



[http://en.wikipedia.org/wiki/Edgar\\_F.\\_Codd](http://en.wikipedia.org/wiki/Edgar_F._Codd)

Codd estableció los fundamentos del modelo relacional en el artículo de 1970 "A Relational Model of Data for Large Shared Data Banks". En adelante, sus publicaciones van ampliando el modelo. Colaborador de aquellos tiempos es Chris Date, del que nos nutrimos en las referencias bibliográficas de la asignatura. Los dos trabajaban para IBM.

## primeras ideas

- Una forma de “ver” los datos
  - Sencillo, intuitivo, potente
  - El más usado actualmente



NUMVEND	NOMVEND	NOMBR...	TELEFONO	CA
200	SEVERINO MARTIN MARTINEZ	SEVESOFT	5779988	GENER
1	AGAPITO LAFUENTE DEL CORRAL	MECEMSA	96-5782401	Avda.
2	LUCIANO BLAZQUEZ VAZQUEZ	HARW S.A.	96-3232321	GENER
3	GODOFREDO MARTIN MARTINEZ	MECEMSA	96-4141722	AVDA.
4	JUANITO REINA PRINCESA	HARW S.A.	903-696969	DONDE
5	JUANITO REINA PRINCESA	LA DEAQUI	98-5363636	S. FRANCISCO DE ASIS, 101 GUION
6	MANOLO PIEDRA POMEZ	HUMP S.A.	96-5660727	AVIACION 92, 3I

NUMPEDIDO	NUMVEND	FECHA
1	1	05/05/92
2	1	11/10/92
3		2 15/10/92
4		2 16/10/92
5	1	22/10/92
6		5 22/08/93
7		8002 02/10/92



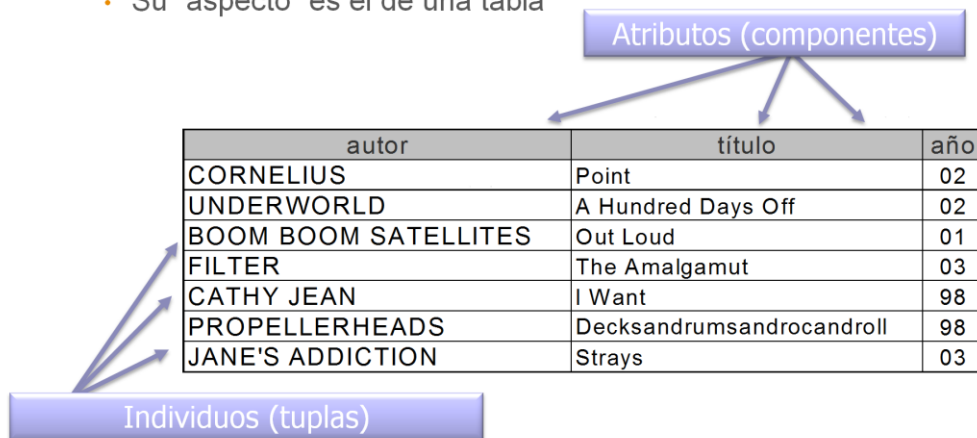
¿Y qué es el modelo relacional?

Una forma de ver los datos, como tablas con filas y columnas (el sistema ya se encarga de almacenar los datos como buenamente pueda, no te preocupes), y que te muestra esos datos, simplemente, preguntando por lo que quieres saber... aunque todo termine en ficheros en un disco duro.

Lo cierto es que esta visión de alto nivel nos oculta, precisamente, los detalles del almacenamiento de datos a bajo nivel (sistema operativo, disco duro, organización y acceso), al tiempo que proporciona una estructura simple basada en conceptos matemáticos hartos conocidos que da coherencia al modelo en sí. Un detalle importante es que Codd da la impresión de querer alejarse todo lo posible de los entornos de programación ofreciendo un método simple e intuitivo de tratar la información. Por eso, desde el punto de vista del usuario normal, la ausencia de punteros o herramientas similares que relacionen datos dispersos, sustituidos por simples comparaciones de valores.

## primeras ideas

- Relación matemática
  - Base formal del modelo
    - Su “aspecto” es el de una tabla



El aspecto final, la percepción que tenemos del almacenamiento de los datos, aunque vamos a ver que es más complejo que eso, es el de una tabla con filas y columnas. Las columnas, al igual que cualquier tabla con cualquier propósito, indican un aspecto concreto, una característica de la clase de objeto que se está representando. Son sus atributos, las características que lo definen dentro de nuestro sistema de información. Son también, como veremos, las componentes de un esquema de tupla, siendo las tuplas el equivalente formal de lo que entendemos por fila. Cada tupla representa a un individuo, un caso, un ejemplo, un objeto que pertenece a la clase de objetos.

En la terminología del paradigma orientado a objeto, la clase de objetos define al objeto, y los objetos son los casos concretos, los elementos que pertenecen o se definen por esa clase de objetos.

## primeras ideas

- SQL
  - *selecciona* "autor", "título"  
*de la tabla* "música"  
*cuando* "año"= 98
  - Opera con tablas y obtiene tablas

autor	título	año
CATHY JEAN	I Want	98
PROPELLERHEADS	Decksanddrumsandrocandroll	98

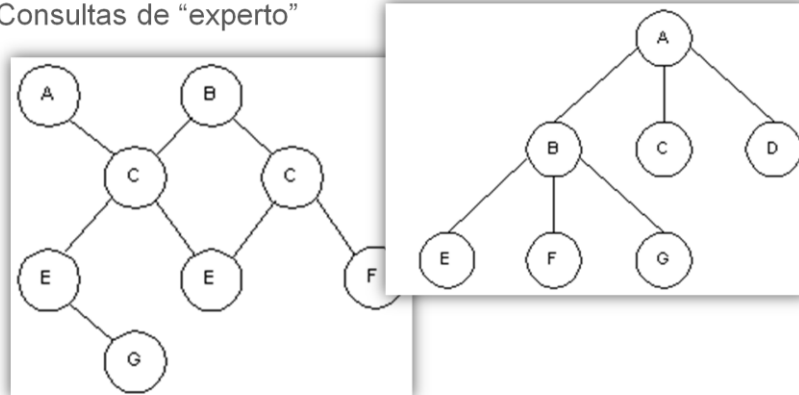


El gran acierto del modelo relacional, y casi seguro el objetivo de Codd al definirlo, es la aparición de un lenguaje que se aparta de los programadores para acercarse a los usuarios con conocimientos mínimos (exagerando, claro, no deja de ser un lenguaje "informático"). El Standard Query Language es un lenguaje de definición y manipulación de datos de especificación, no se programa el "cómo" recuperar los datos sino que se explicita el "qué" se quiere conseguir.

Es un lenguaje cerrado, esto es, opera con, y obtiene, tablas.

## primeras ideas: predecesores

- modelo jerárquico y modelo en red (CODASYL)
  - En realidad, sistemas de ficheros sofisticados (para la época)
    - Dificultad de representación
    - Consultas de "experto"



Antes del modelo relacional, lo más utilizado en grandes volúmenes de datos era sistemas de ficheros sofisticados basados en topologías fácilmente reconocibles pero, también, difíciles de manejar y comprender. De hecho, el mantenimiento de los datos era tarea exclusiva de programadores, llegándose a la situación de que el más mínimo listado debía ser solicitado puesto que se tenía que programar el código que recorriera la estructura de la forma concreta que exigía un determinada lista de datos ordenada y filtrada por ciertos criterios.

El modelo jerárquico estructura los datos en forma de árboles, conectando los registros intermedios y los de hoja mediante punteros. Su capacidad de representación estaba limitada prácticamente a relaciones 1:N, siendo complicado llegar a simular relaciones N:M. El modelo en red (CODASYL, por el comité encargado de definirlo) pretendía superar al jerárquico mediante una estructura menos rígida pero también más difícil de mantener.

## la relación

- Estructuras del modelo
  - Son las “palabras” de este “lenguaje”
  - Conocer las estructuras es conocer el “lenguaje” y poder “hablar”
  - Dictan cómo puedo representar la realidad
    - Cómo se combinan las “palabras” hasta representar conceptos complejos con los que poder comunicarnos con un SGBD y con otras personas



La "relación" es la estructura, única, del modelo relacional. Para entendernos, la "relación" es lo mismo que la tabla (hablando informalmente). En cualquier caso, tal y como establecimos en el tema Modelos de Datos, las estructuras son las "palabras" de estos lenguajes peculiares.

Entiéndase que este es un símil muy forzado. En realidad estamos hablando de una estructura vacía que tenemos que "rellenar" con aquello que consideremos conveniente, y no podemos hacerlo de cualquier manera sino siguiendo ciertas reglas: toda tabla ha de tener al menos una columna, esa columna debe trabajar con un cierto conjunto de datos posibles, debe tener al menos una clave primaria, para relacionarla con otras tablas se definen claves ajenas, etc., hasta llegar a la representación completa de nuestro sistema de información. Y no dará lo mismo poner una clave ajena en una u otra tabla, hacerlo daría como resultado sistemas de información distintos.

# la relación: definición formal

## Relación matemática

$$R \subseteq D_1 \times D_2 \times \dots \times D_n$$

## Producto cartesiano

$$D_1 \times D_2 \times \dots \times D_n = \{ \langle d_1, d_2, \dots, d_n \rangle \mid d_1 \in D_1, d_2 \in D_2, \dots, d_n \in D_n \}$$

## Dominio

Conjunto de valores escalares



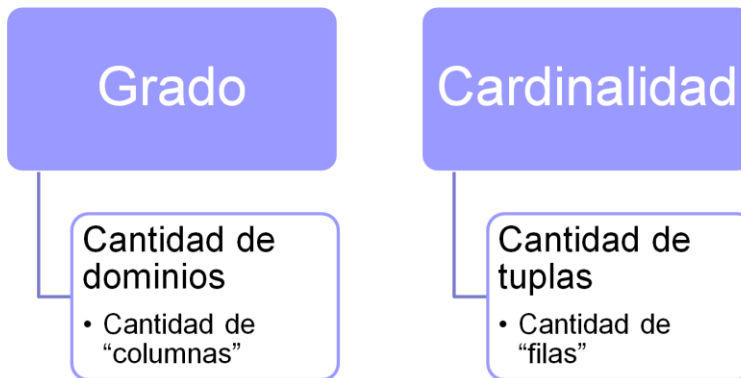
Un dominio es un conjunto de valores escalares. Por valor escalar entendemos un valor simple, sin estructura interna. El nombre de una calle es un dato escalar, la dirección completa (calle, número, código postal, etc.) es un dato complejo.

Se puede operar con varios dominios mediante un producto cartesiano. El producto cartesiano obtiene un conjunto de tuplas; las tuplas tienen tantas componentes como dominios se han utilizado en el producto, y estas componentes ocupan un orden determinado. El resultado final es la combinación de todos los valores de cada dominio con todos los valores de los otros dominios.

Se define la relación matemática como el subconjunto de un producto cartesiano de  $n$  dominios.



## la relación: métricas



***Alumno**  $\subseteq$   $n^{\circ}$  expediente  $\times$  dni  $\times$  nombre  
 $\times$  titulación  $\times$  curso  $\times$  grupo*

Dos métricas simples se asocian a la relación matemática: el grado, la cantidad de dominios que definen el producto cartesiano, y por ende la relación, y la cardinalidad, que es la cantidad de tuplas que conforman la propia relación.

Hablando en términos de filas y columnas, el grado es la cantidad de columnas de la tabla y la cardinalidad la de filas. El grado es constante (salvo que se redefina la relación) mientras que la cardinalidad es variable, depende de en qué momento examinemos la relación habrá más o menos tuplas (habremos insertado o borrado).

Hacemos notar que el término "cardinalidad" aquí tiene un sentido diferente al de las "cardinalidades" que hemos venido usando para concretar las relaciones 1:1, 1:N, N:M, etc. (aunque, en el fondo, están relacionados). Igualmente, "relación" como relación matemática (como "tabla") es un sentido diferente al de "relación" como asociación entre clases de objetos (entidades o tablas).

## la relación: ejemplo

$$R \subseteq D_1 \times D_2$$

- $D_1 = \{ 1, 0 \}$
- $D_2 = \{ a, b, c \}$

$$R = \{ \langle 1, a \rangle \langle 1, b \rangle \langle 1, c \rangle \langle 0, a \rangle \langle 0, b \rangle \langle 0, c \rangle \}$$

- $\text{Grado}(R) = 2$
- $\text{Cardinalidad}(R) = 6$

$$R = \{ \langle 1, a \rangle \langle 1, b \rangle \langle 0, a \rangle \langle 0, b \rangle \}$$

- $\text{Grado}(R) = 2$
- $\text{Cardinalidad}(R) = 4$



Aquí tenemos un ejemplo de una relación definida a partir del producto cartesiano de 2 dominios. Como vemos, en dos instantes consecutivos, el grado se mantiene constante mientras que la cardinalidad cambia.

## la relación: consecuencias

$$R \subseteq D_1 \times D_2$$

**R es un conjunto**

**No** contiene ni contendrá **tuplas** duplicadas

**No** existe orden entre las tuplas

Sí existe orden entre las componentes de las tuplas

El producto nunca "produce" dos tuplas iguales

En los conjuntos no hay relación de orden

Lo define el propio producto, y es la única forma de acceder a una componente



En todo caso, hemos y estamos hablando de conjuntos (<http://es.wikipedia.org/wiki/Conjunto>). El resultado de un producto cartesiano es un conjunto de tuplas. Las tuplas son listas de componentes. La relación es un subconjunto.

Como tales, tienen unas propiedades particulares que Codd iba buscando como base para definir su modelo relacional.

## la relación: adaptación

- Hasta el momento solo tenemos

**Alumno**  $\subseteq \text{domE} \times \text{domD} \times \text{domN}$

1	2	3
3	21333555	LUISA
1	22444666	PEPE
2	21777333	ANA

*¿select 3 from alumno?*



A pesar de lo elegante del formalismo matemático elegido para definir el modelo de datos, no es cómodo acceder a las componentes de la tupla por su orden, debo saber que donde yo estoy almacenando el nombre de los alumnos es la tercera componente de cada tupla. Si le damos nombre a las componentes nos evitamos recurrir a esta ordenación al tiempo que nos resulta más fácil recordar cómo recuperar los datos necesarios.

## la relación: adaptación

- Definición de una relación en el MR

- Esquema* (intensión de R)

$$\mathbf{R} = \{ \langle A_i : D_i \rangle \} \quad i = 1, 2, \dots, n$$

- Contenido* (extensión de R)

$$\mathbf{R} = \{ \{ \langle A_1 : V_{11} \rangle \dots \langle A_n : V_{n1} \rangle \} \\ \dots \\ \{ \langle A_1 : V_{1j} \rangle \dots \langle A_n : V_{nj} \rangle \} \} \\ j = 1, 2, \dots, m$$

- Grado(R) = n
- Cardinalidad(R) = m



Este "dar nombre a las componentes de las tuplas de una relación" se define como se muestra aquí:

- 1) el esquema de la relación (la intensión, la cabecera) es un conjunto de pares (nombre, dominio), de nombres (o etiquetas) y dominios asociados a cada nombre.
- 2) el contenido de la relación (la extensión, el cuerpo) es un conjunto de listas de pares (nombre, valor).

## la relación: adaptación

- Definición de una relación en el MR

- Esquema* (intensión de R)

**Alumno={ <exp:domE> <dni:domD> <nombre:domN> }**

- Contenido* (extensión de R)

**{ { <exp : 3> <dni: 21333555> <nombre : LUISA> }  
{ <nombre : PEPE> <dni: 22444666> <exp : 1> }  
{ <dni: 21777333> <exp : 2> <nombre : ANA> } }**

- Grado(R) = n
- Cardinalidad(R) = m



Ahora nos da igual el orden de las componentes, alumno se define a partir de exp, dni y nombre, cada componente "trabajando" sobre unos conjuntos de valores determinados (domE, domD y domN).

La extensión de la relación, en un momento dado, contendrá m tuplas, y cada tupla será n pares (nombre, valor), y no necesariamente presentaremos todas y cada una de las tuplas con el mismo orden de componentes puesto que está perfectamente definido con qué dato específico estamos trabajando en todo momento.

## la relación: las consecuencias de la adaptación

$$R \subseteq D_1 \times D_2$$

**R es un conjunto**

**No** contiene ni contendrá **tuplas** duplicadas

**No** existe orden entre las tuplas

**No** existe orden entre las componentes de las tuplas

El producto nunca "produce" dos tuplas iguales

En los conjuntos no hay relación de orden

Si no nos interesa, pero también se puede usar



Esta adaptación se traduce en que ya no necesitamos el orden de las componentes (el orden de las columnas). No obstante (y SQL hace uso de ello, por ejemplo ... order by 1 es ordenar por la primera columna), sí se mantiene el orden para ciertas operaciones, como se verá en álgebra relacional. Podemos decir que es una opción: podemos utilizar este orden o no, según nos convenga.

## la relación: abstracción

- Las "herramientas" del MR
  - Clasificación
    - definir atributos y asignarles **dominios**
      - restricciones de dominio + **valor nulo**
  - Agregación
    - definir **relaciones**
      - restricciones de identificador: **claves candidatas**
      - restricciones de correspondencia entre clases: **claves ajenas**
  - Generalización
    - agregación especial de relaciones





## la relación: "implementación"

- **NULL/NULO**= ignorancia
  - No sé si hay o no valor y, si lo hubiere, no sé cuál es
  - difícil implementación
    - se simplifica demasiado a menudo con "ausencia de valor"
    - *¡no es "cadena vacía" o "blanco"!, estos son valores*
- **clave candidata**
  - sea  $R(A:D_1, B:D_2, \dots, Z:D_n)$
  - $\{A_i:D_i\} \subseteq \text{intensión}(R)$  es clave candidata si
    - tiene valores distintos para cada tupla
    - es irreducible
- **clave ajena**
  - correspondencia entre clases
    - relaciones entre clases de objetos



Al final del desarrollo matemático, de alguna forma hemos de trasladar esas intenciones a un producto en funcionamiento. Basándose en la relación matemática, se crean los siguientes conceptos que, finalmente, darán forma al modelo. Todos ellos están viéndose de manera más o menos informal en la segunda hora de las sesiones de teoría.

El modelo relacional se implementa con algo parecido a los registros. Solo con los registros no puedo asegurar que no haya dos iguales, los ficheros no restringen su contenido al exacto producto cartesiano. ¿Cómo conseguir que esta estructura tan simple y típica de ficheros se comporte como un conjunto?: definiendo el concepto de clave candidata y obligando a que siempre se defina para toda relación. Ahora bien, para ser clave candidata no podemos permitir que admita nulos en ninguno de sus atributos. Recordemos que NULO significa que no sabemos qué valor tiene y, si no lo sabemos, podría ser un duplicado y ya no cumpliría con su función.

¿Y cómo facilitamos la estructuración de los datos y sus relaciones? Con las claves ajenas.

## la relación: "implementación"

- Toda relación tiene siempre al menos una clave candidata
  - la totalidad de atributos de la relación cumple la función de identificación
- *Integridad de clave*
  - ningún atributo de una clave candidata puede contener valores nulos
- *Integridad referencial*
  - una clave ajena es totalmente nula o contiene un valor previamente almacenado en la clave primaria a la que hace referencia



Pongamos por ejemplo la relación ALUMNO: **NO** necesitamos todas las posibles combinaciones de DNI y nombre y apellidos, la realidad es que el DNI nunca se va a duplicar, por consiguiente ALUMNO es un subconjunto de  $\text{DNI} \times \text{NOMBRE} \times \text{APELLIDOS}$ . Siendo así, no necesito los 3 atributos para diferenciar a un alumno de otro.

Siguiendo con el mismo ejemplo, tengamos en cuenta que la propiedad de no duplicidad del DNI es externa a la definición de la relación. Dicho en otras palabras, nosotros sabemos que no todas las posibles combinaciones de valores que nos da el producto cartesiano son posibles (además, no todos los números de DNI estarán en mi base de datos, algunos ya han muerto, otros no han nacido, otros simplemente "no los conozco"). En realidad, estamos definiendo un subconjunto del producto cartesiano con una propiedad peculiar: las tuplas elegidas nunca repiten un DNI (cosa que sí pasa en el producto cartesiano).

## la relación: reflexión

- Cuestiones

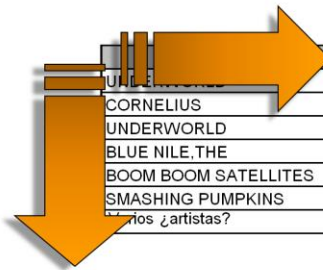
- ¿Por qué se ha elegido el producto cartesiano?
- ¿Por qué la relación es un subconjunto?
- ¿Por qué necesitamos definir al menos una clave candidata?
- ¿Por qué estamos seguros de que existe la clave candidata?
  - Entonces, ¿qué pasa con la dependencia de identificador?



Para definir un subconjunto primero hay que definir "el conjunto": si partimos de todas las posibles combinaciones de valores del producto cartesiano, la relación es un subconjunto del mismo pero **NO ES HABITUAL QUE SEA IGUAL** (aunque en ciertos y raros casos podría ser igual) al producto cartesiano.

Aún así, sí es cierto que siempre podré encontrar al menos una clave candidata: la totalidad de los atributos cumple la condición de ser capaz de identificar a cada tupla, ya que el producto cartesiano no produce duplicados.

# percepción y realidad



	titulo	año
JULIAN ARREST	A Hundred Days Off	02
CORNELIUS	Point	01
UNDERWORLD	Beacoup Fish	98
BLUE NILE, THE	A Walk Across The Rooftop	83
BOOM BOOM SATELLITES	Out Loud	01
SMASHING PUMPKINS	Gish	91
¿quien son los artistas?	Operación Pastelazo	

la tabla sugiere orden entre las filas

la tabla sugiere orden entre las columnas  
(lo que no es incorrecto del todo)

y, además, puedo duplicar filas y columnas



La tabla, pintada en un papel sugiere cosas que no son ciertas, la relación siempre se comportará como tal y mantendrá las restricciones, los límites, propios de la relación. Ni hay orden entre las filas, ni se pueden duplicar, ni hay orden entre las columnas (como ya se ha dicho, si no queremos).

## percepción y realidad

Término formal

- **RELACIÓN**
- atributo
- tupla
- grado
- cardinalidad

Término informal

- **TABLA**
- columna
- fila
- cantidad de columnas
- cantidad de filas



A partir de ahora, lenguaje formal e informal se usarán indistintamente, ya que conocemos perfectamente de qué estamos hablando en todo momento.

## conclusión

- El modelo relacional es el primero con una base formal matemática robusta
- La estructura de partida es muy simple: la relación matemática
  - es un conjunto formado a partir de un producto cartesiano
  - el producto cartesiano se define a partir de dominios, conjuntos de valores escalares
    - los tipos de datos (MySQL, por ejemplo) son casos particulares de los dominios

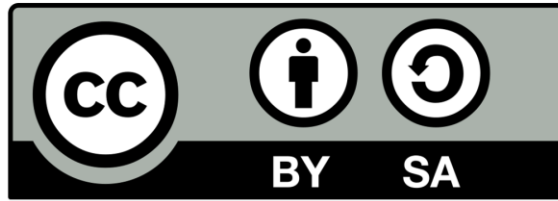


## conclusión: referencias

<http://fbddocs.dlsi.ua.es/lecturas>



## licencias



Todos los logotipos y marcas registradas mostrados en este sitio son propiedad de sus respectivos propietarios y NO están bajo la licencia mencionada.