



**Departamento de Lenguajes y Sistemas Informáticos**

Avda Reina Mercedes s/n. 41012 Sevilla  
Tlf/Fax 954 557 139 E-mail [lsi@lsi.us.es](mailto:lsi@lsi.us.es) [www.lsi.us.es](http://www.lsi.us.es)



# **Bases de Datos**

## **Modelo relacional de Codd**

### **Estructuras y restricciones**

Sevilla, marzo 2004  
V 2004.01.1

<b>1</b>	<b>INTRODUCCIÓN.....</b>	<b>3</b>
1.1	RESEÑA HISTÓRICA.....	3
1.2	SITUACIÓN ACTUAL.....	3
1.3	EVOLUCIÓN DEL MODELO RELACIONAL .....	4
<b>2</b>	<b>ESTRUCTURAS .....</b>	<b>5</b>
2.1	TERMINOLOGÍA .....	5
2.1.1	Relación. ....	5
2.1.2	Dominio. ....	5
2.1.3	Atributo. ....	5
2.1.4	Tupla. ....	6
2.1.5	Grado. ....	6
2.1.6	Cardinalidad. ....	6
2.1.7	Relaciones vs tablas .....	7
2.2	CONTEXTO MATEMÁTICO DEL MR.....	8
2.2.1	Concepto de relación matemática.....	8
2.2.2	Concepto de relación en el modelo relacional.....	8
<b>3</b>	<b>RESTRICCIONES.....</b>	<b>10</b>
3.1	CLAVES EN EL MODELO RELACIONAL .....	10
3.1.1	Concepto intuitivo de identificador .....	10
3.1.2	Concepto de superclave .....	10
3.1.3	Clave Candidata.....	10
3.1.4	Clave Primaria.....	11
3.1.5	Clave Alternativa .....	11
3.1.6	Clave Ajena .....	11
3.2	REGLAS DE INTEGRIDAD DEL MR.....	12
3.2.1	Valor “null” en el MR .....	12
3.2.2	Integridad de la entidad.....	13
3.2.3	Integridad referencial.....	13

# 1 Introducción

## 1.1 Reseña histórica

En 1970 Edgar F. Codd publicó un trabajo<sup>1</sup> proponiendo un nuevo modelo de datos que perseguía como objetivo general la **flexibilidad y sencillez**; el modelo ofrece estructuras de datos simples y lenguajes sencillos para los usuarios, facilitando la comprensión y utilización por estos. Sin menoscabo de este objetivo de sencillez, el autor se propone fundamentar el modelo sobre una sólida base matemática.

La aparición del modelo relacional representa un verdadero hito en el desarrollo de las bases de datos, ya que ha marcado tres etapas diferentes, conocidas como generaciones de los SGBD's:

- **Pre-relacionales.** Los SGBD se basan en modelos Codasyl (en red) y Jerárquico y ficheros planos (flat files).
- **Relacionales.** Los sistemas relacionales ganan madurez en el mercado y los productos basados en este modelo van desplazando poco a poco a los sistemas basados en punteros de la etapa pre-relacional.
- **Post-relacionales.** Aparecen manifiestos de otros modelos de datos, en especial los orientados a objeto. Se distinguen manifiestos puristas OO que dan lugar a SGBDs-OO puros como O2, Gemstone, etc. y, en paralelo, corrientes evolutivas del modelo relacional que relajan hipótesis básicas del modelo original de Codd (relajación de la primera forma normal) para ofrecer estructuras de datos más complejas. Se propone una evolución desde el modelo relacional a SGBDs-OO relacionales, p. ej. SQL3.

Sobre el modelo relacional se han definido los estándares ANSI e ISO del extendido lenguaje de definición y manipulación de bases de datos relacionales SQL (Structured Query Language).

## 1.2 Situación actual

Desde mediados de los años 80, el modelo relacional es utilizado por prácticamente la totalidad de los SGBD comerciales.

- Algunas de las principales empresas informáticas del mundo, son en origen, empresas fabricantes de SGBDs relacionales: ORACLE, Sybase, INFORMIX, etc.
- Existen grandes fabricantes de hardware y software que tienen "su" SGBD relacional: IBM: DB2, Microsoft: SQL Server, etc.
- Existen SGBDs diseñados para PC's y usuarios no expertos: Microsoft Access, etc.

---

<sup>1</sup> "A relational model for large shared data banks"; publicado en Communications of the ACM.

### 1.3 Evolución del modelo relacional

P R E R E L A C I O N A L		1968 - 1970	↔	Surge el modelo
		1970 ...	↔	Desarrollos teóricos
		1973 - 1978	↔	Prototipos (Ingres, sistema R, etc. . .)
	R E L A C I O N A L	1978	↔	QBE
		1979	↔	Oracle
		1980	↔	Ingres
		1981	↔	SQL
		1982	↔	DB2
		1986	↔	SQL/ ANS
		1987	↔	SQL ISO (9075)
		1989	↔	SQL Addendum
		1989	↔	Manifiesto de los SGBO
		1990	↔	Modelo Relacional Versión 2
P O S T R E L A C I O N A L		1990	↔	Manifiesto de los SGBO- 3G
		1992	↔	SQL 92
		1995	↔	3 <sup>er</sup> Manifiesto
		1999	↔	SQL 3

*Fig.1 Evolución del modelo relacional.*

## 2 Estructuras

### 2.1 Terminología

#### 2.1.1 Relación.

Es la estructura básica del modelo relacional. Con una relación es posible representar tanto instancias de una entidad del universo real como interrelaciones entre entidades de distinto tipo. Es capaz de recoger interrelaciones de cardinalidad múltiple. Su representación informal es una tabla.

#### 2.1.2 Dominio.

Es el conjunto válido de valores de referencia para definir propiedades o atributos.

Un dominio es un conjunto nominado y homogéneo de valores.

Cada dominio puede definirse de dos maneras:

- Por **extensión** (enumeración de sus elementos):  
días de la semana = {lunes, martes, miércoles, jueves, viernes, sábado, domingo}
- Por **intensión** (mediante una propiedad que recoja el recorrido de sus valores admisibles):  
edad: entero /  $0 \leq \text{edad} \leq 200$ .

Un **dominio compuesto** se puede definir como una combinación de dominios simples a la que se puede aplicar ciertas restricciones.

Ej.: el dominio compuesto denominado **Fecha** se construye por agregación de los dominios simples **Día**, **Mes** y **Año**, incorporando las restricciones a fin de que no aparezcan valores inválidos como: 29/2/2003, 31/4/2004.

#### 2.1.3 Atributo.

Representa las propiedades de la relación. Un atributo, necesariamente ha de definirse sobre un dominio. Su representación informal es una columna.

Un atributo (A) es la interpretación de un determinado dominio en una relación, es decir el “papel” que juega el dominio en la misma.

**$D = \text{Dom}(A) \Rightarrow D$  es el dominio de A**

- Un atributo está siempre asociado a una relación, mientras que un dominio tiene existencia propia con independencia de las relaciones que existan en el modelo.
- Un atributo representa una propiedad de una relación.
- Un atributo toma valores de un dominio.
- Varios atributos distintos (de la misma o de diferentes relaciones) pueden tomar sus valores del mismo dominio.

### 2.1.4 Tupla.

Es una ocurrencia o instancia dentro de una relación. Una tupla permite referenciar una instancia de una entidad en el universo o la interrelación específica o concreta entre instancias de entidades. Su representación informal es una fila. Una relación tiene un “conjunto” de tuplas.

La relación es el elemento fundamental del modelo relacional.

NOMBRE	atributo 1	atributo 2	.....	atributo n	
	XXX	XXX	.....	XXX	→ tupla 1
	XXX	XXX	.....	XXX	→ tupla 2
	.....	.....	.....	.....	.....
	XXX	XXX	.....	XXX	→ tupla m

Fig.2 Representación de una relación en forma de tabla

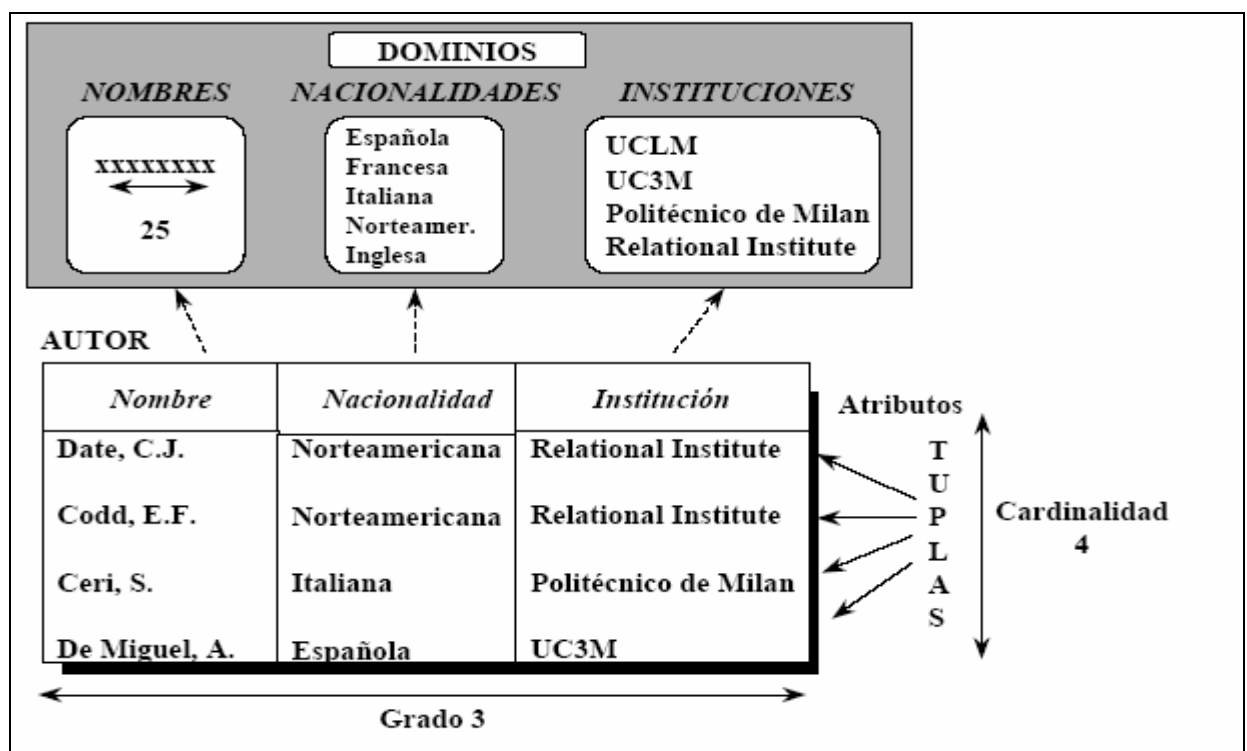


Fig.3 Representación de la relación autor

### 2.1.5 Grado.

Se define como el número de dominios  $D_i$

### 2.1.6 Cardinalidad.

Se define como el número de tuplas de la relación.

### 2.1.7 Relaciones vs tablas

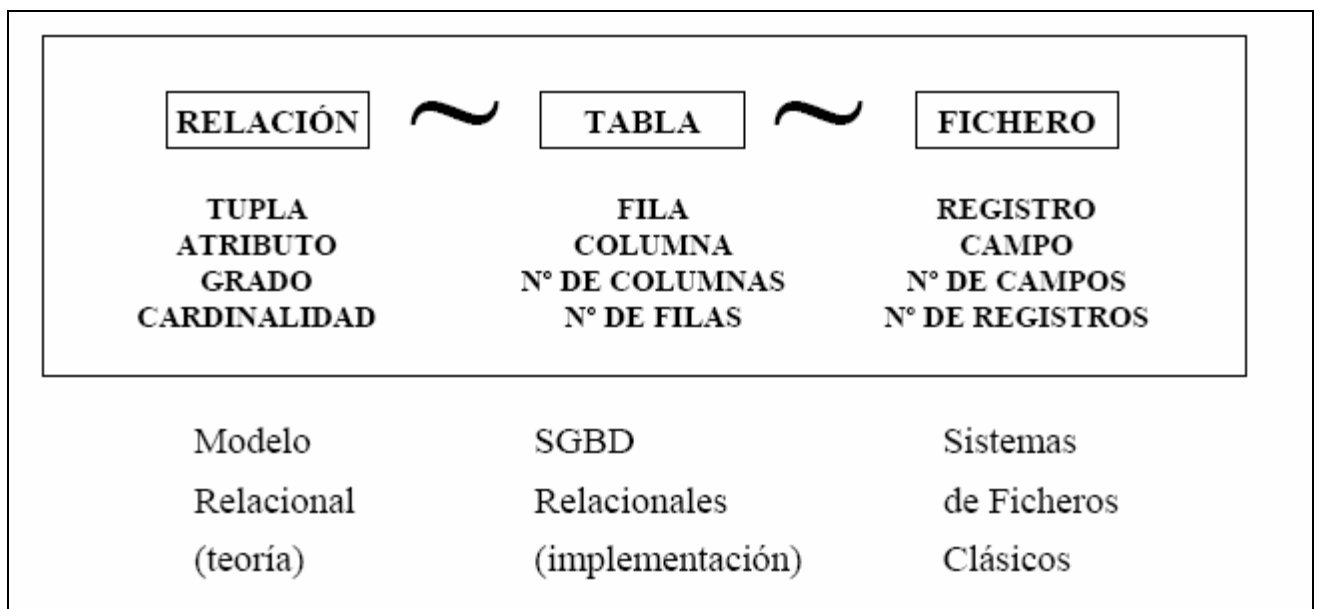
Existen diferencias entre ambas estructuras.

#### Tabla

- Un orden definido para las filas.
- Un orden definido para los valores de los atributos.
- Pueden repetirse filas.

#### Relación

- Ni las tuplas ni los valores de los componentes de una tupla se referencian por su posición relativa (no hay un orden predefinido para el conjunto).
- No pueden repetirse tuplas, pues un conjunto no tiene elementos repetidos.



*Fig. 4 Terminología*

## 2.2 Contexto matemático del MR

### 2.2.1 Concepto de relación matemática

Puede definirse una relación apoyándose en la teoría de conjuntos:

Dados los dominios  $D_1 \dots D_n$ , no necesariamente distintos, el producto cartesiano  $D^* = D_1 \times D_2 \times \dots \times D_n$  es el conjunto de todas las posibles n-tuplas:

$$t < d_1, d_2, \dots, d_n > / \forall i \Rightarrow d_i \in D_i$$

$R$  es una relación definida sobre  $D^*$  si es un subconjunto del producto cartesiano  $D^* : R \subseteq D^*$

Cada elemento de la relación ( $t$  *tupla*) es una lista de n valores, tales que el valor i-ésimo corresponde a una propiedad que toma valores del dominio i-ésimo.

El número de tuplas de la relación recibe el nombre de **Cardinalidad** y el número de dominios  $D_i$  el nombre de **Grado**

Una relación es un conjunto, luego:

- No existen tuplas duplicadas
- Las tuplas no están ordenadas.

Esta definición conjuntista de relación ofrece una **granularidad<sup>2</sup> a nivel de tupla**. La granularidad a nivel de tupla ofrece una fuerte dependencia entre estructuras de datos y programas, de modo que los cambios en la estructura del esquema de una relación puede obligar a modificar los programas de aplicación que acceden a dicha relación.

### 2.2.2 Concepto de relación en el modelo relacional

Con objeto de conseguir una granularidad más fina y mejorar la independencia de datos, Codd propone relajar la hipótesis de correspondencia posicional de valores de atributos en una tupla respecto a sus dominios. Se redefine una relación como:

- Un nombre de relación.
- Un conjunto de atributos  $A = (\{A_i\})$  donde el número de atributos es el grado de la relación.
- Un conjunto de dominios  $D = (\{D_i\})$ .
- Una aplicación que asocia a cada atributo de A un dominio D. **Dom:  $A \rightarrow D$**

Estos cuatro componentes de una relación reciben el nombre de **intensión o esquema de la relación** y se representa con la notación  $R (\{A_i: D_i\})$

- Un conjunto de tuplas  $\{t\} / \forall t \exists \mathfrak{T} : v_i \rightarrow A_i$  de modo que  $t \equiv \{A_i : v_i\}$   
 $\{t\}$  se denomina **extensión de la relación**.

Con esta definición no hay una relación posicional de valores/dominios, sino una correspondencia atributo/valor dentro de cada tupla; se ha conseguido granularidad a nivel de atributo elemental. Se recuperan los valores en una tupla por la denominación del atributo en vez de por su posición. Los cambios

---

<sup>2</sup> Se entiende por granularidad el nivel de acceso a un gránulo de datos. Ej. de gránulos: fichero, registro, valor.



en el esquema de la relación que impliquen adicionar nuevos atributos no afectarán a programas de aplicación que accedían a un subconjunto de valores de una tupla.

<b>ESQUEMA DE RELACIÓN (INTENSIÓN):</b>		
<b>AUTOR</b> ( <i>Nombre: Nombres, Nacionalidad: Nacionalidades, Institución: Instituciones</i> )		
<b>RELACIÓN (EXTENSIÓN, ESTADO u OCURRENCIA):</b>		
<b>AUTOR</b>		
<i>Nombre</i>	<i>Nacionalidad</i>	<i>Institución</i>
<b>Date, C.J.</b>	<b>Norteamericana</b>	<b>Relational Institute</b>
<b>De Miguel, A.</b>	<b>Española</b>	<b>UC3M</b>
<b>Ceri, S.</b>	<b>Italiana</b>	<b>Politécnico de Milan</b>

*Fig. 5 Intensión y extensión de una relación*

## 3 Restricciones

### 3.1 Claves en el modelo relacional

#### 3.1.1 Concepto intuitivo de identificador

Al manejar cualquier estructura de datos se hace necesario distinguir elementos en conjuntos de datos. Intuitivamente, un identificador, en general, es un subconjunto de propiedades que permite distinguir o identificar elementos dentro de dicho conjunto.

En general, pueden existir distintos identificadores en una estructura de datos.

#### 3.1.2 Concepto de superclave

En el modelo relacional, como una relación es un conjunto, no puede tener elementos duplicados, luego está implícita la existencia de, al menos un identificador para las tuplas de la relación.

Una superclave cumple la condición de **unicidad**:

Sea  $R(\{A_i : D_i\})$

$SK = \{A_j\} \subseteq R$

$SK$  es una superclave de  $R$  si  $\nexists t_1, t_2 / (t_1 \neq t_2) \wedge (t_1.K = t_2.K)$

#### 3.1.3 Clave Candidata

Sea  $R(\{A_i : D_i\})$

$K = \{A_j\} \subseteq R$

$K$  es una clave candidata si cumple dos condiciones:

I) Unicidad

$\nexists t_1, t_2 / (t_1 \neq t_2) \wedge (t_1.K = t_2.K)$

II) Minimalidad

$\nexists K' \subset K, K' \neq \emptyset / K'$  verifique la condición (I) de unicidad en  $R$

De la definición de relación se deriva que siempre existe, al menos, una clave candidata (al ser una relación un conjunto y no existir dos tuplas iguales, el conjunto de todos los atributos siempre tiene que identificar unívocamente a cada tupla).

El concepto de clave es muy relevante en el modelo relacional y representa el mecanismo para referenciar las tuplas de una relación.

### 3.1.4 Clave Primaria

Una relación puede tener más de una clave candidata.

La clave primaria (*Primary Key*) es la clave candidata que el usuario escoge para identificar las tuplas de la relación (siempre existe clave primaria).

### 3.1.5 Clave Alternativa

Las claves alternativas (*Alternative Key*) son las claves candidatas que no han sido escogidas como clave primaria.

**ESCRITOR ( Nombre, Apellido1, Apellido2, Dni, Nacionalidad, Institución)**

**Pk** (Dni)

**Ak** (Nombre, Apellido1, Apellido2)

**Claves candidatas** (Dni) y (Nombre, Apellido1, Apellido2)

Ej. de **superclave**: (Dni, Nacionalidad)  
(Dni)  
(Dni, Nombre, Apellido1, Apellido2)  
(Nombre, Apellido1, Apellido2, Institución)

No son superclaves: (Nombre, Apellido1)  
(Nacionalidad, Institución)

Fig. 6 Claves candidatas, primaria, alternativa y superclave

### 3.1.6 Clave Ajena

Sean:  $R(\{A_i : D_i\})$  y  $S(\{B_j : E_j\})$  no necesariamente distintas, donde

$PK_R = \{A_{it}\}$  y  $PK_S = \{A_{jt}\}$  son las claves primarias de cada relación

Si  $\exists FK_{R/S} = \{A'_{it}\} / (FK_{R/S} \subset R, FK_{R/S} \neq \emptyset)$   
 $\wedge (dom(FK_{R/S}) = dom(PK_S))$

$FK_{R/S}$  es una clave ajena en la relación **R** respecto de la relación **S**

Las **claves ajenas** son el mecanismo de referencia del MR. Permite representar interrelaciones 1:1 y 1:m incluyendo la clave primaria de la relación referenciada en la relación desde que se hace la referencia. También es posible representar interrelaciones múltiples (m:n) definiendo una nueva relación cuya clave primaria se compone de las claves primarias de las relaciones referenciadas, actuando, a la vez, dichas claves como claves ajenas de dichas relaciones.

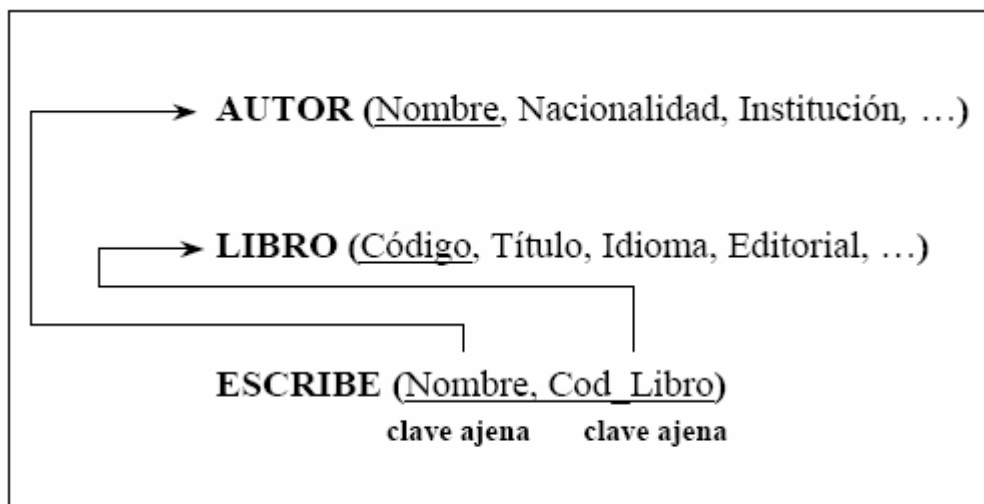


Fig. 7 Claves primarias y ajenas

Con estas definiciones pueden enunciarse las dos reglas de integridad explícitas del modelo relacional.

## 3.2 Reglas de integridad del MR

### 3.2.1 Valor “null” en el MR

Se define como **null** el valor que puede tomar un atributo en una tupla cuando se desconoce dicho valor; p.ej. “No se sabe el domicilio de una persona”, “Se desconoce su fecha de nacimiento, etc.”. Se diferencia el valor **null** de cualquier otro valor (blanco,--, etc.) que implique una asignación a un atributo; **null** significa que dicho atributo no tiene asociado ningún valor o que “**su valor es desconocido**”. **Null** puede asignarse a atributos en cualquier dominio; ahora bien, no es equivalente el valor **null** de un atributo tipo fecha con el valor **null** del domicilio de una persona; es decir no se pueden comparar nulos de atributos distintos; sí puede evaluarse si el valor de un atributo es o no **null**.

**Alumnos**(dni:D\_dni, nombre:String, domicilio: String, Fecha\_nac:Date) **Pk**(dni)

Ej. de tuplas:

$t_1$ <28645789V, “Juan López Castro”, “Avda. Reina Mercedes 24-5º”, #1/12/1982#>  
 $t_2$ <75476123K, “Ana Sánchez Benjumea”, **null**, #1/10/1981#>  
 $t_3$ <33444444F, “Araceli Valdivieso Hernández”, “San Jacinto 83 -3º”, **null**>  
 $t_4$ <07802444S, “Antonio Márquez Sánchez”,**null**, **null**>

### 3.2.2 Integridad de la entidad

La regla de integridad de la entidad es el mecanismo que gobierna el comportamiento de la clave primaria de una relación, de modo que la representación de entidades en el modelo relacional garantice la identificación o distinción de las mismas en el universo de modelado.

*"Ningún atributo que forme parte de la clave primaria de una relación puede tomar un valor nulo"*; es decir, un valor desconocido o inexistente.

Sea  $R(\{A_i:D_i\})$   $P_k = \{A_{it}\} \subseteq R$

$t \in R \rightarrow (\forall A_{it} \rightarrow t.A_{it} \neq \text{null})$

#### Justificación de la regla:

- I) **Si  $P_k$  es un atributo elemental**, p.ej. DNI del alumno, éste no puede tomar el valor null, puesto que si la propiedad es desconocida, significa que en el mundo modelado, no se conoce ¿Quién es el alumno? Y por tanto tampoco podría distinguirse de otro alumno; esto independientemente de que sólo haya una tupla con ese valor.
- II) **Si  $P_k$  está formada por un descriptor compuesto**, P.ej. (tienda, dni:cliente, fecha\_pedido) como subconjunto de atributos elegido para la identificar **pedidos** en una tienda realizados por un cliente; si se estima que en algún caso, es decir en alguna tupla, con la (tienda y el dni:cliente) puede identificarse dicha tupla es que, en general, puede desecharse la **fecha\_pedido** para identificar las tuplas del pedido, con lo que esto contradice que (tienda, dni:cliente, fecha\_pedido) sea clave primaria, pues no cumple la condición de minimalidad; (tienda, dni:cliente, fecha\_pedido) sería una superclave.

### 3.2.3 Integridad referencial

La regla de integridad referencial gobierna las situaciones de representación de interrelaciones en el universo de discurso, de modo que se representan vínculos existentes o inexistentes, pero se eviten referencias no permitidas.

Sean

$R(\{A_i:D_i\})$  y  $S(\{B_j:E_j\})$  no necesariamente distintas donde

$PK_R = \{A_{it}\}$  y  $PK_S = \{A_{jt}\}$  son las claves primarias de cada relación

Si  $\exists FK_{R/S}$  y es una clave ajena en la relación R respecto de la relación S, entonces:

I) Existe una referencia en R respecto a S:

$[(t \in R) \wedge (t.FK \neq \text{null})] \rightarrow \exists s \in S / s.PK = t.FK$

o bien

II) No se referencia ninguna tupla de S:

$[(t \in R) \wedge (\exists A'_{it} \subseteq FK / t.A'_{it} = \text{null})] \rightarrow \forall A'_{it} \subseteq FK, t.A'_{it} = \text{null} \rightarrow \nexists s \in S / s.PK = t.FK$

### Justificación de la regla:

- I) **Si FK referencia una tupla**, p.ej. editorial del libro, si toma un valor no nulo, este valor necesariamente ha de corresponder a una tupla de editorial existente en la relación de editoriales. No se admiten un valor que no esté debidamente referenciado.
- II) **Si FK no referencia una tupla**,  
 Ej.I) El valor del código de departamento de un profesor es nulo; esto significaría que el profesor no está asignado a ningún departamento (si este supuesto semántico tiene cabida en el universo modelado).  
 Ej.II) El descriptor (NºPedido,LíneaPedido,LíneaFactura) es la clave primaria de la relación **Líneas\_de\_factura** (NºPedido,LíneaPedido,LíneaFactura,cant,precio) y la clave ajena (NºPedido,LíneaPedido) sobre una relación de líneas de pedido toma el valor (104,null). No es posible incluir una línea de factura sin referenciar una línea de pedido aunque se referencia el código del pedido. Es necesario definir la tupla (NºPedido,LíneaPedido) que, a su vez, es la clave primaria de la relación Líneas\_de\_pedido referenciada.  
 Ej.III)

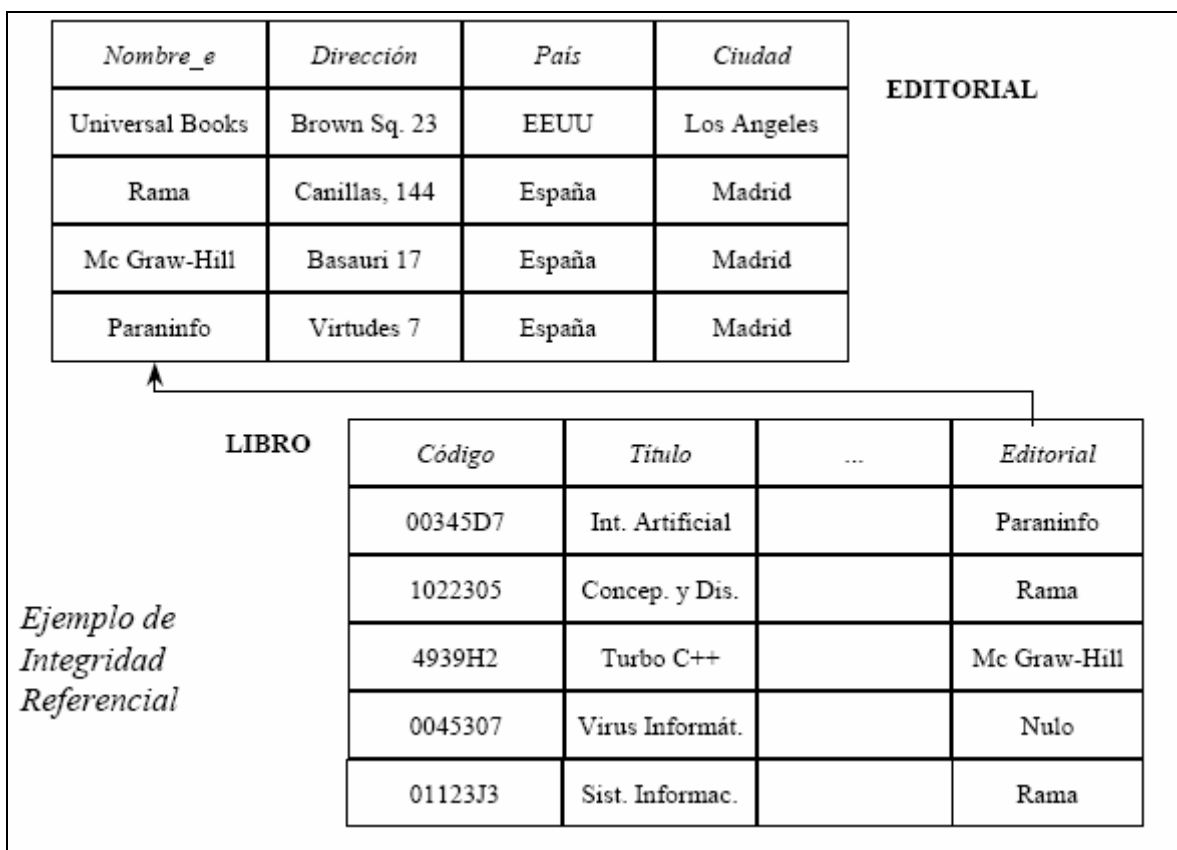


Fig. 8 Integridad referencial