

## **Tema 3: El modelo relacional.**

### **3.1. Evolución del modelo relacional.**

Las desventajas de los modelos jerárquico y en red condujeron a un intenso interés en el nuevo modelo de datos relacional propuesto por Codd en 1970. Este modelo constituye un intento de simplificar las estructuras de las bases de datos.

El nuevo modelo propuesto por Codd se basaba en la teoría de las relaciones, donde los datos se estructuran en forma de tablas o relaciones. Los objetivos perseguidos por Codd en su modelo son los siguientes, objetivos también comunes a otros modelos:

- Independencia física, es decir, que la manera en que estén almacenados físicamente los datos en los soportes de almacenamiento no incida en la manipulación lógica de los datos y que, por tanto, no sea necesario cambiar los programas porque cambie el almacenamiento físico de los datos.
- Independencia lógica, es decir, que al añadir eliminar o modificar objetos en la base de datos no se vean afectados los programas y/o usuarios que estén accediendo a subconjuntos de los datos.
- Uniformidad, consistente en que los datos presentan una estructura lógica con un aspecto uniforme, lo que facilita la concepción y manipulación de la base de datos por parte de los usuarios.
- Sencillez: Todo lo indicado anteriormente junto con lenguajes de usuario sencillos, dan como resultado un modelo fácil de entender y utilizar para el usuario final.

### **3.2. Estructura del modelo relacional.**

En una base de datos relacional los datos se almacenan en relaciones y una relación se puede representar por medio de una tabla.

#### **3.2.1. El concepto de relación. Propiedades de las relaciones.**

Toda relación tiene un nombre y consta de un conjunto de filas y columnas. Las columnas se corresponden con los atributos de la relación o propiedades de la misma. Por su parte, las filas se llaman también tuplas y cada tupla contiene una serie de valores para cada uno de los atributos de la relación.

El número de columnas de una relación se llama grado y el número de filas, cardinalidad.

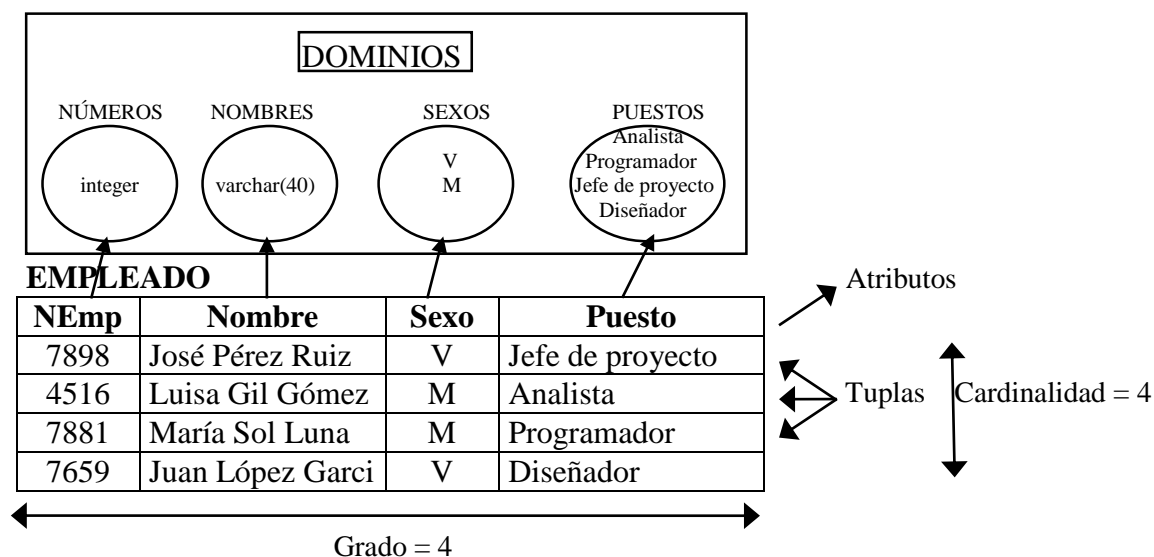


Figura 1: Ejemplo de relación.

La cabecera de una relación es la parte estática de la misma, es decir, aquella que se ve modificada muy de vez en cuando, y consta de un conjunto de atributos con sus dominios subyacentes. Un dominio es el conjunto de valores permitidos para un atributo. Por ejemplo, el dominio para el atributo *Sexo* se puede llamar *Sexos* e incluye solo dos posibles valores: V (que hace referencia a varón) y M (que hace referencia a mujer). El dominio para el atributo *Nombre* se puede llamar *Nombres* y hace referencia a cualquier cadena de hasta 40 caracteres.

Por su parte, el cuerpo de la relación es la parte dinámica de la misma y está constituido por una serie de tuplas que van variando con el paso del tiempo a medida que los usuarios introducen, eliminan y modifican datos.

Podemos decir que una relación tiene las siguientes propiedades:

- Toda relación tiene un nombre único dentro de la base de datos relacional a la que pertenece.
- Los valores de los atributos de una relación son atómicos, lo que quiere decir que en la intersección de una fila y una columna hay un solo valor.
- Todos los atributos de una tabla tienen nombres distintos. Sin embargo, puede haber atributos de distintas tablas de una misma base de datos con el mismo nombre.
- Los atributos de una relación no están ordenados.
- En una tabla no hay dos tuplas iguales.
- Las tuplas de una relación no están ordenadas.

### 3.2.2. Atributos y dominio de los atributos.

Como se ha indicado en la sección anterior, los atributos son cada una de las propiedades de una relación y cada atributo constituye una columna de la tabla por medio de la cual se representa una relación. Así, la relación *Pedido* de la siguiente figura consta de dos atributos llamados *RefPed* y *FecPed*:

PEDIDO	
RefPed	FecPed
P0001	16/02/2024
P0002	18/02/2024
P0003	23/02/2024
P0004	25/02/2024

Figura 2: Relación *Pedido*.

Como se indicó también en la sección anterior, cada atributo tiene asignado un único dominio, que hace referencia al conjunto de valores permitidos para dicho atributo. Así, para el caso de la relación *Pedido* de la figura 2, el dominio del atributo *RefPed* es una cadena de 5 caracteres y el dominio para el atributo *FecPed* es cualquier fecha válida.

### 3.2.3. Tupla, grado y cardinalidad.

Como se indicó en la sección 3.2.1, cada fila de una relación también recibe el nombre de tupla, por lo podemos decir que la relación *Pedido* de la figura 2 consta de cuatro tuplas.

El grado de una relación hace referencia a su número de atributos, por lo que la relación *Pedido* tiene grado 2, mientras que su cardinalidad o número de filas es 4.

### 3.2.4. Relaciones y tablas.

El hecho de que una relación se represente por medio de una tabla es lo que ha originado el que en muchos casos se utilice el término tabla para hacer referencia a relación, columna para hacer referencia a atributo y fila para hacer referencia a tupla.

No obstante, es interesante tener en cuenta que aunque una relación se pueda representar en forma de tabla, tiene una serie de elementos característicos que la distinguen de una tabla, ya que en una relación no se permiten tuplas duplicadas y en una tabla sí que puede haber filas repetidas. Además, las filas y las columnas no están ordenadas y en el cruce de una fila y una columna solo puede haber un valor (no se admiten atributos multivaluados).

### 3.3. Claves en el modelo relacional.

En el modelo relacional se puede hablar de distintos tipos de claves, que se exponen en las siguientes subsecciones.

#### 3.3.1. Claves candidatas.

Se define clave candidata de una relación como un conjunto de atributos que identifican unívoca y mínimamente a cada tupla de la relación. El que identifican unívocamente a cada tupla quiere decir que no va a poder haber en una relación varias tuplas con el mismo valor para esa clave candidata. La condición de minimalidad hace referencia a que el conjunto de atributos es el más pequeño posible, es decir, si un atributo ya permite identificar a cada tupla, no tiene sentido que la clave candidata esté formada por ese atributo y algún otro. En la relación de la figura 1 el atributo *NEmp* es clave candidata ya que no puede haber dos empleados con el mismo valor para este atributo. Si además supiésemos que no va a haber dos empleados con el mismo nombre, entonces el atributo *Nombre* también sería clave candidata. Por su parte, la relación *Pedido* de la figura 2 solo tiene una clave candidata, que es el atributo *RefPed*, porque nunca va a haber dos pedidos con la misma referencia.

Consideremos a continuación la relación *Artículo* cuyo contenido se muestra en la siguiente figura.

ARTÍCULO		
CodArt	DesArt	PVPArt
A0043	Bolígrafo azul fino	0,78
A0078	Bolígrafo rojo normal	1,05
A0075	Lápiz 2B	0,55
A0012	Goma de borrar	0,15
A0089	Sacapuntas	0,25

Figura 3: Relación *Artículo*.

En esta relación una clave candidata será *CodArt* porque nunca va a haber dos artículos con el mismo código. Si, además, nunca va a poder haber dos artículos con la misma descripción, el atributo *DesArt* también será clave candidata.

#### 3.3.2. Claves primarias.

Se puede definir clave primaria como aquella clave candidata elegida para identificar las tuplas de la relación. Cuando solo existe una clave candidata, esta será la clave primaria. Por ejemplo, en la relación *Empleado* de la figura 1, podríamos elegir *NEmp* como clave primaria, aunque si el nombre de los empleados fuera único, también podríamos elegir

*Nombre* como clave primaria. En la relación *Pedido* de la figura 2, será *RefPed* la clave primaria por ser la única clave candidata. Por su parte, en la relación *Artículo* de la figura 3, si *DesArt* fuese clave candidata, podríamos seleccionar como clave primaria tanto *CodArt* como *DesArt*, pero solo uno de ellos.

Las relaciones también se suelen representar indicando su nombre y a continuación los atributos de que constan entre paréntesis y separados por comas. La clave primaria de una relación se suele indicar o bien subrayando el/los atributo/s que constituye/n la clave primaria o bien anteponiendo a dichos atributos el símbolo #. Así, las relaciones de las figuras 1, 2 y 3 se podrían representar de las dos siguientes formas:

Empleado (NEmp, Nombre, Sexo, Puesto)  
 Pedido (RefPed, FecPed)  
 Artículo (CodArt, DesArt, PVPArt)

Figura 4: Representación de relaciones.

Empleado (#NEmp, Nombre, Sexo, Puesto)  
 Pedido (#RefPed, FecPed)  
 Artículo (#CodArt, DesArt, PVPArt)

Figura 5: Representación alternativa de relaciones.

### 3.3.3. Claves alternativas.

Son claves alternativas aquellas claves candidatas que no han sido escogidas como clave primaria. En la relación *Empleado* de la figura 1, el atributo *Nombre* sería una clave alternativa si no pudiese haber varios empleados con el mismo nombre. En la relación *Pedido* de la figura 2, no habría claves alternativas y en la relación *Artículo* de la figura 3, sería clave alternativa *DesArt* si la descripción de los artículos fuese única.

### 3.3.4. Claves ajenas.

Una clave ajena o clave foránea es un conjunto de atributos de una relación cuyos valores deben coincidir con los valores de la clave primaria de otra relación. Debe tenerse en cuenta que la clave ajena y la correspondiente clave primaria deben estar definidas sobre el mismo dominio.

Consideremos que en la misma base de datos en la que están incluidas las relaciones *Pedido* y *Artículo*, disponemos de una tercera relación llamada *LíneaPedido*, que contiene las

líneas de pedido de los pedidos realizados por los clientes. Una línea de pedido es la parte de un pedido correspondiente a cada artículo diferente solicitado en el mismo. Cada línea de pedido vendrá, por tanto, identificada por el pedido (atributo *RefPed*) y el artículo solicitado (atributo *CodArt*). Además, en esta relación se almacenará también un tercer atributo, llamado *CantArt*, que indica la cantidad o número de unidades que del artículo en cuestión se solicita en el pedido correspondiente. Esta relación, que tendrá como clave primaria la pareja de atributos *RefPed* y *CodArt*, la podemos representar de la siguiente forma y podemos considerar que puede almacenar los datos indicados a continuación:

LíneaPedido (RefPed, CodArt, CantArt)

**LINEAPEDIDO**

<b>RefPed</b>	<b>CodArt</b>	<b>CantArt</b>
P0001	A0043	10
P0001	A0078	12
P0002	A0043	5
P0003	A0075	20
P0004	A0012	15
P0004	A0043	5
P0004	A0089	50

Figura 6: Relación *LineaPedido*.

Pues bien, en la tabla *LineaPedido* el atributo *RefPed* es una clave ajena al atributo *RefPed* de la tabla *Pedido*. Por este motivo, el valor que tome el atributo *RefPed* en la tabla *LineaPedido* deberá coincidir con uno de los cuatro valores que toma el atributo *RefPed* en la tabla *Pedido* (P0001, P0002, P0003 o P0004).

Asimismo, el atributo *CodArt* de la tabla *LineaPedido* es una clave ajena al atributo *CodArt* de la tabla *Artículo*. Por este motivo, el valor que tome este atributo en la tabla *LineaPedido* deberá coincidir con uno de los cinco valores que toma el atributo *CodArt* en la tabla *Artículo* (A0043, A0078, A0075, A0012 o A0089).

Las claves ajenas se representan trazando flechas desde cada clave ajena hasta la correspondiente clave primaria. Por ello, la base de datos relacional formada por las tres tablas referidas hasta el momento, se representaría de la siguiente forma:

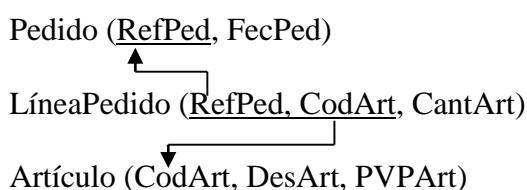


Figura 7: Esquema relacional.

### 3.4. Restricciones de integridad.

Las principales restricciones de integridad del modelo relacional son las siguientes:

- Clave primaria (PRIMARY KEY): permite declarar un atributo o un conjunto de atributos como clave primaria de una relación, por lo que sus valores no se podrán repetir ni se admitirán los valores nulos (o inexistentes).
- Unicidad (UNIQUE), mediante la cual se indica que los valores de un atributo o conjunto de atributos no pueden repetirse en una relación. Esta restricción permite la definición de claves alternativas. Por ejemplo, en la relación *Articulo*, si no puede haber dos artículos con la misma descripción (atributo *DesArt*), este atributo será una clave alternativa y, por tanto, deberá tener asignada la restricción de unicidad.
- Obligatoriedad de uno o más atributos (NOT NULL), con lo que se indica que un atributo o conjunto de atributos no admite valores nulos.
- Integridad referencial (FOREIGN KEY): a esta restricción se hace referencia en la sección 3.4.3.
- Restricciones de rechazo y aserciones: para crear estas restricciones el usuario especifica un predicado o condición sobre un atributo o conjunto de atributos de manera que cada vez que se lleve a cabo una operación de actualización sobre la base de datos, se comprueba si se cumple el predicado especificado y en caso de que este no se cumpla, la operación no es permitida. Podemos hablar de dos tipos de restricciones: las restricciones de verificación o de rechazo, en las que la condición o predicado afecta a una sola relación; y las aserciones, en las que el predicado afecta a varias relaciones.

#### 3.4.1. Valores nulos en el modelo.

Un valor nulo es una marca que indica que el dato asociado a un atributo es desconocido. Los valores nulos son útiles a la hora de manejar información que presenta lagunas, es decir, valores que sí existen en el universo del discurso, pero que se desconocen en un momento dado. Por ejemplo, se están introduciendo las tuplas correspondientes a una tabla de clientes y no se conoce el teléfono de un cliente determinado.

Otro caso en el que se suelen usar los valores nulos es cuando para cierta condición del universo del discurso un atributo determinado no tiene valor. Por ejemplo, si a partir de una serie de albaranes se genera una factura, mientras el proceso de generación de facturas no se lleve a cabo, la clave ajena *NumFactura* en la relación *Albarán* deberá permanecer con valor nulo.

En otro caso, normalmente, cuando se crea una nueva columna en una tabla existente, los valores de la columna añadida para todas las tuplas deben iniciarse con valor nulo.

El tratamiento de los valores nulos es complejo y requiere de un estudio pormenorizado para las distintas operaciones y funciones que se pueden llevar a cabo en el modelo relacional. De este modo, por ejemplo, se puede definir una lógica trivaluada para contemplar la existencia de valores nulos a la hora de determinar el resultado de las operaciones lógicas AND, OR y NOT. Las tablas de verdad correspondientes a estos operadores serían las siguientes:

<b>AND</b>	<b>VERDADERO</b>	<b>FALSO</b>	<b>NULO</b>
<b>VERDADERO</b>	VERDADERO	FALSO	NULO
<b>FALSO</b>	FALSO	FALSO	FALSO
<b>NULO</b>	NULO	FALSO	NULO

<b>OR</b>	<b>VERDADERO</b>	<b>FALSO</b>	<b>NULO</b>
<b>VERDADERO</b>	VERDADERO	VERDADERO	VERDADERO
<b>FALSO</b>	VERDADERO	FALSO	NULO
<b>NULO</b>	VERDADERO	NULO	NULO

<b>NOT</b>	
<b>VERDADERO</b>	FALSO
<b>FALSO</b>	VERDADERO
<b>NULO</b>	NULO

Asimismo, conviene introducir un nuevo tipo de operador que se puede denominar MAYBE (quizás) y que corresponde a la pregunta ¿ES NULO?, cuya tabla de verdad sería:

<b>MAYBE</b>	
<b>VERDADERO</b>	FALSO
<b>FALSO</b>	FALSO
<b>NULO</b>	VERDADERO

Para las operaciones aritméticas, el resultado de operar un valor nulo con otro valor cualquiera se considera siempre nulo.

Hay que tener en cuenta que en las operaciones de selección es posible que no aparezcan tuplas por presentar valores nulos para la condición o el predicado establecido, ya que, si el resultado de la evaluación del predicado es nulo, la tupla correspondiente no aparecerá en el resultado.

Para los operadores de comparación surgen problemas serios, por ejemplo, si nos planteamos si 3 es mayor que, menor que o igual a un valor nulo cualquiera, la única respuesta posible es quizás porque un valor nulo es desconocido.



En cuanto a la presencia de valores nulos en claves, se ha de cumplir lo siguiente:

- Una clave primaria no puede presentar valores nulos por la regla de integridad de la entidad, a la que se hace referencia en la sección 3.4.2.
- Una clave alternativa, al tratarse de una clave candidata, no debería contener valores nulos, pero la mayoría de los SGBD no lo implementan de este modo, siendo necesario asignar una restricción de obligatoriedad (NOT NULL) a los atributos que constituyen claves alternativas.
- Una clave ajena puede contener valores nulos.

En cualquier caso, la recomendación es ser muy restrictivos a la hora de permitir la presencia de nulos, limitando esta posibilidad preferentemente a los casos en los que la semántica del universo del discurso pueda presentar valores de este tipo (por ejemplo, claves ajenas que aún no han podido migrar, fechas que todavía no tienen sentido, etc.).

### 3.4.2. Integridad de las entidades.

Existe una restricción inherente al modelo relacional, que es la regla de integridad de la entidad, la cual establece que: “Ningún atributo que forme parte de la clave primaria de una relación puede tomar un valor nulo”.

### 3.4.3. Integridad referencial.

La restricción de integridad referencial establece que una clave ajena solo podrá tomar valor nulo, o bien alguno de los valores que toma la clave candidata de la relación apuntada.

Así, para las relaciones *Pedido*, *Artículo* y *LineaPedido* expuestas en la sección 3.3.4, el atributo *RefPed* de la relación *LineaPedido* es clave ajena que referencia al atributo homónimo de la relación *Pedido*, de modo que sus valores deben o bien ser nulos o bien concordar con los que toma el atributo *RefPed* de la relación *Pedido* (P0001, P0002, P0003 o P0004). Para este caso en concreto, no es posible que el atributo *RefPed* en *LineaPedido* tome valor nulo al formar este atributo parte de la clave primaria de la relación. Ocurre exactamente lo mismo con la otra clave ajena de la relación *LineaPedido* (su atributo *CodArt*). Este no podrá tomar valor nulo, sino solo uno de los valores que toma el atributo *CodArt* en la relación *Articulo* (A0043, A0078, A0075, A0012 o A0089).

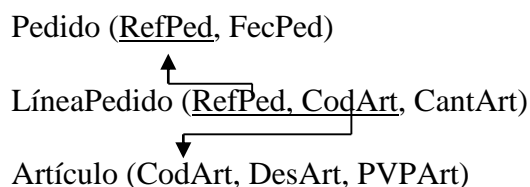


Figura 8: Esquema relacional.

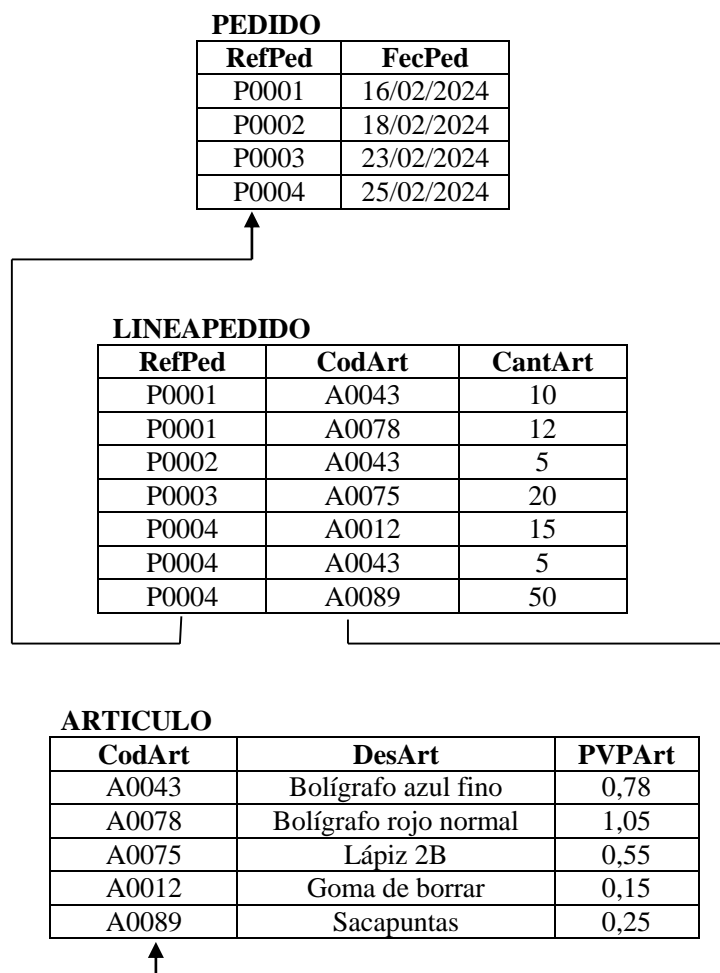


Figura 9: Esquema relacional con datos y claves ajenas.

Además de definir las claves ajenas, hay que determinar las consecuencias que tienen el borrado y modificación de tuplas de la relación referenciada, pudiéndose distinguir las siguientes opciones:

- Operación restringida (RESTRICT): no se va a permitir el borrado o modificación de tuplas de la relación referenciada si hay alguna tupla en la otra relación que contiene el mismo valor en la clave ajena. Aplicado a la primera clave ajena de nuestro ejemplo, implicaría que para borrar un pedido de la relación *Pedido* no puede haber ninguna fila en la relación *LineaPedido* para ese mismo pedido. Además, no se podría modificar el

atributo *RefPed* para un pedido de la tabla *Pedido* para el que hubiese alguna línea de pedido.

- Operación con transmisión en cascada (ON DELETE CASCADE, ON UPDATE CASCADE): el borrado o modificación de tuplas de la relación que contiene la clave referenciada implica el borrado o modificación en cascada de las tuplas correspondientes en la tabla que contiene la clave ajena. Esto implicaría que al borrar un pedido se eliminasen automáticamente todas sus líneas de pedido de la tabla *LineaPedido*. Asimismo, al modificar el atributo *RefPed* de un pedido de la tabla *Pedido*, se modificaría el atributo *RefPed* en la tabla *LineaPedido* para todas las líneas de pedido correspondientes a ese pedido.
- Operación con puesta a nulos (SET NULL): el borrado o modificación de tuplas de la relación que contiene la clave referenciada lleva consigo poner a valor nulo el atributo que constituye la clave ajena. Esto nos llevaría a que, si se borra un pedido de la tabla *Pedido*, se asigne el valor nulo al atributo *RefPed* para todas las líneas de pedido correspondientes a ese pedido. Esta opción, en este caso, no está permitida porque el atributo *RefPed* en la relación *LineaPedido* no puede tomar valor nulo al formar este atributo parte de la clave primaria. Además, si se modifica el atributo *RefPed* para algún pedido en la tabla *Pedido*, se debería asignar valor nulo al atributo *RefPed* en todas las tuplas de la tabla *LineaPedido* correspondientes a líneas de pedido de ese pedido. Ocurre, como en el caso anterior, que esta opción tampoco es válida por el motivo ya indicado.
- Operación con puesta a valor por defecto (SET DEFAULT): se actúa de igual modo que en la anterior opción con la salvedad de que en vez de asignar al atributo que constituye la clave ajena el valor nulo, se le asigna un valor establecido por defecto de antemano para dicho atributo.
- Operación que desencadena un procedimiento de usuario: en este caso, el borrado o modificación de tuplas de la tabla referenciada provoca la ejecución de un procedimiento definido por el usuario.

Para cada clave ajena sería necesario indicar la opción seleccionada. En el ejemplo propuesto habría que indicarlo para las dos claves ajenas existentes en la tabla *LineaPedido* (*RefPed* y *CodArt*). Además, la opción seleccionada en caso de borrado es independiente de la seleccionada en caso de modificación.

### 3.5. Álgebra relacional.

El álgebra relacional es un lenguaje formal de consultas sobre bases de datos relacionales que constituye la base del lenguaje de consultas SQL ampliamente utilizado.

El álgebra relacional se inspira en la teoría de conjuntos para especificar consultas en una base de datos relacional. Para especificar una consulta es preciso definir uno o más pasos que sirven para ir construyendo, mediante operadores del álgebra relacional, una nueva relación que contenga los datos que responden a la consulta a partir de una serie de relaciones. Los operadores del álgebra relacional toman una o más relaciones como operandos y producen otra relación como resultado.

#### 3.5.1. Clasificación de los operadores.

Los operadores del álgebra relacional se pueden clasificar atendiendo a diversos criterios:

- Según se parecen o no a los operadores de la teoría de conjuntos, se clasifican en:
  - Operadores conjuntistas u operadores tradicionales de la teoría de conjuntos: son la unión, la intersección, la diferencia y el producto cartesiano.
  - Operadores relacionales especiales: son la selección, la proyección, la combinación y la división.
- Según se puedan expresar o no en términos de otros operadores, se clasifican en:
  - Operadores primitivos: son los operadores esenciales que no se pueden obtener de otros, es decir, aquellos a partir de los cuales se puede definir el resto. Estos operadores son la selección, la proyección, el producto cartesiano, la unión y la diferencia.
  - Operadores derivados: se pueden obtener aplicando varios de los operadores primitivos, por lo que podríamos prescindir de ellos; sin embargo, son muy útiles y simplifican muchas operaciones habituales. Estos operadores son la intersección, la combinación y la división.
- Según el número de relaciones que tienen como operando:
  - Operadores unarios: son aquellos que tienen una sola relación como operando. Estos operadores son la selección y la proyección.
  - Operadores binarios: son aquellos que tienen dos relaciones como operandos. Estos operadores son el producto cartesiano, la unión, la diferencia, la intersección, la combinación y la división.

Para todos los operadores conjuntistas con excepción del producto cartesiano (unión, intersección y diferencia) las dos relaciones operandos deben ser compatibles con respecto a la unión, es decir, deben ser del mismo grado (deben tener el mismo número de atributos) y los atributos que ocupan las mismas posiciones deben tener el mismo dominio subyacente.

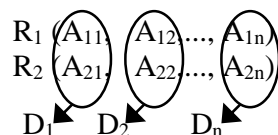


Figura 10: Relaciones compatibles con respecto a la unión.

En los ejemplos que se van a realizar para las operaciones unión, intersección y diferencia, se van a emplear las siguientes relaciones:

**Empleados1**

NumEmp	NomEmp	Puesto
1234	José Pérez Ruiz	Presidente
4543	Luisa Gil Gómez	Vendedor
2323	María Sol Luna	Director

**Empleados2**

NumEmp	NomEmp	Puesto
2344	Mar García Gil	Analista
3423	Mario Ros Bueno	Vendedor
4543	Luisa Gil Gómez	Vendedor

Figura 11: Relaciones de ejemplo para unión, intersección y diferencia.

En los ejemplos que se van a realizar para las operaciones de selección, proyección, producto cartesiano y combinación, se van a emplear las siguientes relaciones.

**Empleados**

NumEmp	NomEmp	Puesto	NumDep
1234	José Pérez Ruiz	Presidente	1
4543	Luisa Gil Gómez	Vendedor	3
2323	María Sol Luna	Director	1

**Departamentos**

NumDep	NomDep	Ciudad
1	Contabilidad	Bilbao
2	Investigación	Santander
3	Ventas	Madrid

Figura 12: Relaciones de ejemplo para selección, proyección, producto cartesiano y combinación.

### 3.5.2. Operaciones primitivas: selección, proyección, producto cartesiano, unión y diferencia.

En esta sección se van a estudiar los operadores primitivos del álgebra relacional:

- Selección o restricción: el operador algebraico de selección ( $\sigma$ ) produce un subconjunto horizontal de una relación dada. Este subconjunto está formado por todas las tuplas de la relación dada para las cuales se cumple un predicado especificado. Por ejemplo, la expresión del álgebra relacional que habrá que usar para seleccionar los empleados del departamento número 1 será la siguiente:

$$\sigma_{\text{NumDep} = 1}(\text{Empleados})$$

El resultado de esta selección será el siguiente:

NumEmp	NomEmp	Puesto	NumDep
1234	Jose Pérez Ruiz	Presidente	1
2323	María Sol Luna	Director	1

Figura 13: Selección sobre una relación.

- **Proyección:** el operador de proyección ( $\pi$ ) produce un subconjunto vertical de una relación dada. Este subconjunto es el obtenido al seleccionar los atributos especificados en el orden indicado de izquierda a derecha, y eliminando luego las tuplas duplicadas. Así, la expresión del álgebra relacional para obtener los puestos de los empleados de la relación *Empleados*, será la siguiente:

$$\pi_{\text{Puesto}}(\text{Empleados})$$

El resultado de esta proyección es el siguiente:

Puesto
Presidente
Vendedor
Director

Figura 14: Proyección sobre una relación.

También se pueden combinar ambos operadores. Así, por ejemplo, la expresión del álgebra relacional para obtener los nombres y puestos de los empleados que trabajan en el departamento número 1, será la siguiente:

$$\pi_{\text{NomEmp, Puesto}}(\sigma_{\text{NumDep} = 1}(\text{Empleados}))$$

El resultado de esta selección y proyección es el siguiente:

NomEmp	Puesto
Jose Pérez Ruiz	Presidente
María Sol Luna	Director

Figura 15: Selección y proyección sobre una relación.

- **Producto cartesiano:** el producto cartesiano de dos relaciones A y B ( $A \times B$ ) es una relación que incluye todas las tuplas posibles que se obtienen concatenando una de A con una de B. Si A es de grado  $m$  y B es de grado  $n$ , la relación  $A \times B$  será de grado  $m+n$ . Si la cardinalidad de A es  $p$  y la de B es  $q$ , la cardinalidad de  $A \times B$  será  $p \times q$ . Así, el producto cartesiano entre las relaciones *Empleados* y *Departamentos*, proporciona la siguiente relación:

**Empleados x Departamentos**

NumEmp	NomEmp	Puesto	NumDep	NumDep	NomDep	Ciudad
1234	José Pérez Ruiz	Presidente	1	1	Contabilidad	Bilbao
1234	José Pérez Ruiz	Presidente	1	2	Investigación	Santander
1234	José Pérez Ruiz	Presidente	1	3	Ventas	Madrid
4543	Luisa Gil Gómez	Vendedor	3	1	Contabilidad	Bilbao
4543	Luisa Gil Gómez	Vendedor	3	2	Investigación	Santander
4543	Luisa Gil Gómez	Vendedor	3	3	Ventas	Madrid
2323	María Sol Luna	Director	1	1	Contabilidad	Bilbao
2323	María Sol Luna	Director	1	2	Investigación	Santander
2323	María Sol Luna	Director	1	3	Ventas	Madrid

Figura 16: Producto cartesiano de dos relaciones.

- Unión: la unión de las relaciones A y B ( $A \cup B$ ) es una relación que incluye todas las tuplas de A y todas las tuplas de B. Si hubiera alguna repetida, se eliminaría la repetida. Así, la unión de las relaciones *Empleados1* y *Empleados2*, daría el siguiente resultado, donde se ha eliminado una de las tuplas correspondientes a la empleada número 4543, por estar repetida.

**Empleados1  $\cup$  Empleados2**

NumEmp	NomEmp	Puesto
1234	José Pérez Ruiz	Presidente
4543	Luisa Gil Gómez	Vendedor
2323	María Sol Luna	Director
2344	Mar García Gil	Analista
3423	Mario Ros Bueno	Vendedor

Figura 17: Unión de dos relaciones.

- Diferencia: la diferencia de las relaciones A y B ( $A - B$ ) es una relación que incluye todas las tuplas que pertenecen a A pero no pertenecen a B. Así, la diferencia entre las relaciones *Empleados1* y *Empleados2* proporciona el siguiente resultado:

**Empleados1 – Empleados2**

NumEmp	NomEmp	Puesto
1234	José Pérez Ruiz	Presidente
2323	María Sol Luna	Director

Figura 18: Diferencia entre dos relaciones.

### 3.5.3. Otras operaciones: intersección, join, división.

En esta sección se van a estudiar diversos operadores derivados del álgebra relacional:

- Intersección: la intersección de dos relaciones A y B ( $A \cap B$ ) es una relación que incluye todas las tuplas que pertenecen a la vez a A y a B. Así, la intersección entre las relaciones *Empleados1* y *Empleados2* produce la siguiente relación, que incluye solo la tupla

correspondiente a la empleada 4543, que es el único empleado que aparece a la vez en las dos relaciones.

<b>Empleados1 <math>\cap</math> Empleados2</b>		
<b>NumEmp</b>	<b>NomEmp</b>	<b>Puesto</b>
4543	Luisa Gil Gómez	Vendedor

Figura 19: Intersección de dos relaciones.

- **Combinación, reunión o join:** el resultado de la combinación de dos relaciones A y B es una relación que incluye todas las tuplas que se obtienen concatenando una de A y otra de B tales que cumplan una determinada condición para los valores de un atributo de dominio común a ambas. Para ello, en primer lugar se realiza el producto cartesiano de las dos relaciones, y a continuación una selección de las tuplas que cumplan la condición especificada. El operador es  $*$ .

Consideremos las relaciones *Empleados* y *Departamentos*. Vamos a combinar estas tablas tal que el atributo *NumDep* de la tabla *Empleados* tome el mismo valor que el atributo homónimo de la tabla *Departamentos*. Esto se expresará de la siguiente forma:

$$(\text{Empleados} * \text{Departamentos})_{\text{Empleados.NumDep} = \text{Departamentos.NumDep}}$$

Para llevar a cabo esta operación es necesario, en primer lugar, realizar el producto cartesiano, cuyo resultado es el siguiente:

#### **Empleados x Departamentos**

NumEmp	NomEmp	Puesto	NumDep	NumDep	NomDep	Ciudad
1234	Jose Pérez Ruiz	Presidente	1	1	Contabilidad	Bilbao
1234	Jose Pérez Ruiz	Presidente	1	2	Investigación	Santander
1234	Jose Pérez Ruiz	Presidente	1	3	Ventas	Madrid
4543	Luisa Gil Gómez	Vendedor	3	1	Contabilidad	Bilbao
4543	Luisa Gil Gómez	Vendedor	3	2	Investigación	Santander
4543	Luisa Gil Gómez	Vendedor	3	3	Ventas	Madrid
2323	María Sol Luna	Director	1	1	Contabilidad	Bilbao
2323	María Sol Luna	Director	1	2	Investigación	Santander
2323	María Sol Luna	Director	1	3	Ventas	Madrid

Figura 20: Producto cartesiano.

A continuación, es preciso quedarse solo con aquellas tuplas para las que se cumpla el predicado especificado, es decir, en este caso, que el atributo *NumDep* de la tabla *Empleados* tome el mismo valor que el atributo *NumDep* de *Departamentos*. El resultado es el siguiente:



**(Empleados \* Departamentos)** Empleados.NumDep = Departamentos.NumDep

NumEmp	NomEmp	Puesto	NumDep	NumDep	NomDep	Ciudad
1234	Jose Pérez Ruiz	Presidente	1	1	Contabilidad	Bilbao
4543	Luisa Gil Gómez	Vendedor	3	3	Ventas	Madrid
2323	María Sol Luna	Director	1	1	Contabilidad	Bilbao

Figura 21: Combinación o reunión.

Cuando la condición, como en este caso, es una comparación de igualdad, se trata de una equijoin, equirreunión o combinación por igualdad.

La combinación más habitual entre tablas es la llamada join o reunión natural, que es el resultado de una equijoin con la eliminación de los atributos duplicados. En la equijoin se igualan los atributos a través de los cuales se establece la relación entre las dos tablas (clave ajena y correspondiente clave primaria). El operador es  $\bowtie$ .

Por tanto, la reunión natural entre dos relaciones  $A$  y  $B$  se lleva a cabo así:

- Se concatenan las tuplas de  $A$  y  $B$  (se calcula  $A \times B$ ).
- Se seleccionan de entre esas tuplas las que tengan iguales valores en las columnas de igual dominio consideradas (clave ajena y correspondiente clave primaria).
- Se suprime una columna de cada dos homónimas (con el mismo nombre) en el resultado, es decir, se eliminan las columnas duplicadas.

La reunión natural entre las tablas *Empleados* y *Departamentos* tiene como dos primeros pasos los mostrados en las figuras 20 y 21. La realización de la reunión natural entre estas dos tablas implica en el resultado de la combinación (figura 21) eliminar una de las dos columnas *NumDep*. El resultado es el siguiente:

**Empleados  $\bowtie$  Departamentos**

NumEmp	NomEmp	Puesto	NumDep	NomDep	Ciudad
1234	Jose Pérez Ruiz	Presidente	1	Contabilidad	Bilbao
4543	Luisa Gil Gómez	Vendedor	3	Ventas	Madrid
2323	María Sol Luna	Director	1	Contabilidad	Bilbao

Figura 22: Combinación natural.

- División: el operador división divide una relación  $A$  de grado  $m$  entre una relación  $B$  de grado  $n$  y produce una relación resultado de grado  $m-n$ . En general  $A$  suele ser de grado 2 y  $B$  de grado 1. La relación  $A/B$  resultado es el conjunto de las tuplas  $\langle x \rangle$  tales que al concatenarlas con las tuplas de  $B \langle y \rangle$  producen tuplas contenidas en  $A \langle x, y \rangle$ .

$$A/B \times B \subseteq A$$

Al hacer el producto cartesiano de  $A/B$  y  $B$ , las tuplas resultantes deben pertenecer a  $A$ .

En el siguiente ejemplo se realiza la división entre la relación *Empleados* de grado 4 y una nueva relación llamada *Depar* de grado 1, que se muestra en la figura 23. La relación

*Empleados/Depar* será de grado 3 e incluirá todos los atributos de *Empleados* excepto los de *Depar* y las filas de empleados que trabajan en el departamento número 1 (el que aparece en la tabla *Depar*).

<b>Depar</b>
<b>NumDep</b>
1

Figura 23: Relación *Depar* para división.

<b>Empleados/Depar</b>		
<b>NumEmp</b>	<b>NomEmp</b>	<b>Puesto</b>
1234	Jose Pérez Ruiz	Presidente
2323	María Sol Luna	Director

Figura 24: División de dos relaciones.