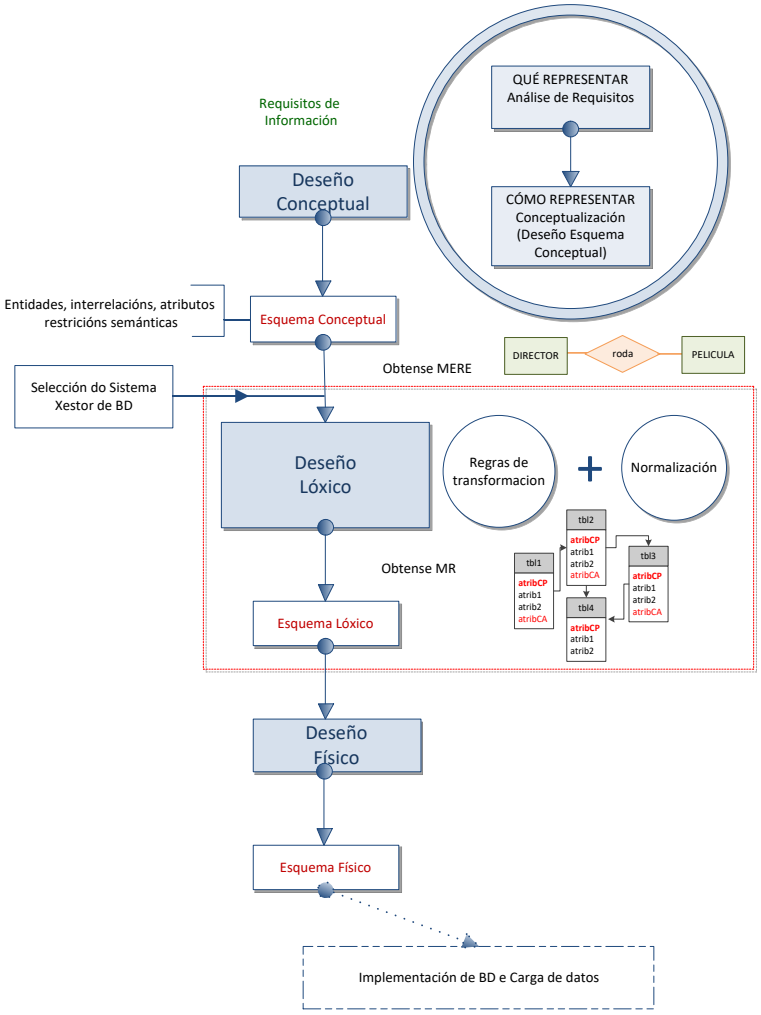
1. Descripción y representación gráfica del modelo relacional

1.1. Diseño lógico

La fase de diseño lógico se desarrolla después de la fase de diseño conceptual, que se representa mediante el diagrama Entidad-Relación. Consiste en aplicar sobre el diseño conceptual unas reglas de transformación que permiten obtener el modelo de datos lógico, que posteriormente se va a utilizar en la fase de diseño físico para crear y manejar la base de datos. La imagen muestra gráficamente esta sucesión de pasos.



El modelo lógico de datos empleado por la mayoría de los SGBD actuales es el modelo relacional (SGBDR). Para este modelo de datos, la fase de diseño lógico produce como resultado un esquema relacional, que representa las tablas que se utilizarán en la base de datos y las relaciones que existen entre ellas; este esquema se representa gráficamente mediante un grafo o un diagrama que pueden estar acompañados de un diccionario de datos y un conjunto de reglas del modelo que no pueden ser representadas gráficamente. El modelo Lógico Relacional debe ser refinado mediante un proceso de Normalización para evitar repeticiones, anomalías o pérdidas de información, para procurar obtener la mayor eficiencia y optimización de funcionamiento.

1.1.1 Fundamentos del modelo relacional

A finales de los años sesenta, E.F. Codd desarrolló el modelo de datos relacional, en los laboratorios de IBM en San José (California). Codd ha propuesto un modelo de datos basado en la teoría de las relaciones, donde los datos se estructuran lógicamente en forma de relaciones (tablas), con el objetivo fundamental de mantener la independencia de esta estructura lógica respecto al modo de almacenamiento y a otras características de tipo físico.

La introducción de la teoría de las relaciones en el campo de las bases de datos ha supuesto un importante paso en la investigación de los SGBD, suministrando un sólido fundamento teórico para el desarrollo de nuevos productos, dentro de este enfoque relacional. La base teórica del modelo Relacional (MR) aparece publicada por Codd en 1970, y propone una representación de la información que:

- Origine esquemas que representen fielmente la información, los objetos y las relaciones que hay entre ellos.
- Pueda ser entendida fácilmente por los usuarios que no tienen preparación previa en esta área.
- Haga posible ampliar el esquema de la base de datos sin modificar la estructura lógica existente y, por lo tanto, sin modificar los programas de aplicación.
- Permita la máxima flexibilidad en la formulación de consultas no previstas.

Los avances más importantes que el modelo de datos relacional incorpora respecto de los modelos de datos que existían antes son:

- Sencillez y uniformidad.

  Las bases de datos relacionales están formadas lógicamente por un conjunto de tablas, lo que les aporta gran uniformidad. Por otro lado, emplean lenguajes no procedimentales que hacen sencillo el manejo de las mismas.

- Sólida base teórica.

  El modelo está definido con rigor matemático, y el diseño y la evaluación del mismo puede realizarse por métodos sistemáticos basados en abstracciones.

- Independencia de la interfaz de usuario.

  Los lenguajes relacionales manipulan conjuntos de registros, por lo que proporcionan una gran independencia respecto de la forma en la que los datos están almacenados.

Las ventajas citadas han contribuido a que desde mediados de los años 80, el MR (Modelo Relacional) sea utilizado por prácticamente la totalidad de los SGBD comerciales, siendo el modelo más utilizado para modelar sistemas reales de aplicaciones comerciales de procesamiento de datos convencionales. Se impuso debido a las limitaciones de los modelos anteriores como el de Codasyl (modelo en red) o el jerárquico y a la falta de un estándar para los modelos orientados a objetos. Ejemplos: algunas de las principales empresas informáticas del mundo son en origen empresas de SGBD (ORACLE, Sybase, INFORMIX, ...); los grandes fabricantes de software tienen "su" SGBD relacional (IBM DB2, Microsoft SQL Server, ...); existen bastantes SGBD diseñados para PC's y usuarios no expertos (Microsoft Access...). Además de los SGBD relacionales de los grandes fabricantes, tienen un gran peso en el mercado algunos productos de software libre como MariaDB, PostgreSQL...

1.1.2 El grafo relacional

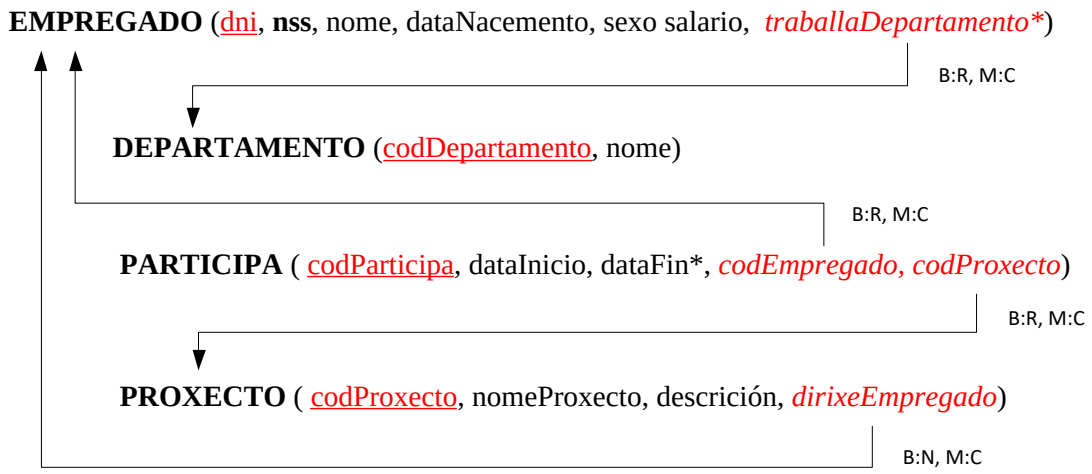
El grafo relacional, también denominado grafo de combinación o diagrama esquemático, es una representación de un sistema mediante un conjunto de relaciones vinculadas entre

sí, empleando una simbología concreta. Es el resultado gráfico de la transformación del esquema conceptual Entidad-Relación al esquema lógico relacional estándar; a partir de él, podrá crearse la BD empleando el SGBDR seleccionado. El grafo relacional incluye información sobre relaciones, atributos, claves primarias, claves candidatas, y claves foráneas.

Existen varias propuestas o versiones para la representación gráfica del grafo relacional. La propuesta que adoptaremos al comienzo de esta unidad es la de Adoración de Miguel y Mario Piattini que tiene las siguientes características:

- Cada relación se representa por un nombre de relación, que se escribe en mayúsculas, y un conjunto de atributos entre paréntesis.
- Los atributos que compongan la clave primaria se mostrarán subrayados. En este documento se marcarán también con color rojo.
- Los atributos que compongan las claves candidatas se mostrarán en negrita.
- Los atributos que compongan la clave foránea se móstrarán en color rojo y estilo de fuente cursiva, y desde éstos se dibujará una línea con punta de flecha que llegue hasta la relación a la que hace referencia.
- Los atributos opcionales, es decir, que podrían tener valores nulos, se representan poniendo un asterisco a continuación de su nombre.

Ejemplo de grafo relacional:



1.1.3 Elementos y terminología del modelo relacional

1.1.3.1 Relación

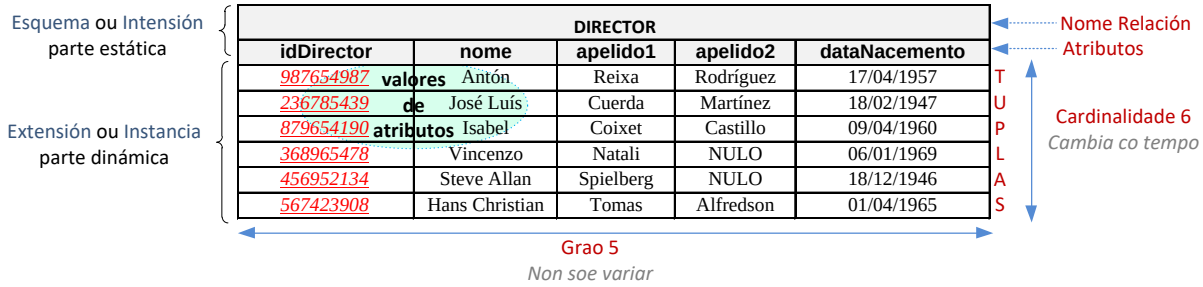
La relación es el elemento básico en el modelo relacional y puede considerarse desde el punto de vista matemático como un conjunto de datos que puede representarse gráficamente como una tabla o matriz rectangular, en la que cada fila se llama tupla y cada columna corresponde a un atributo. Una relación queda definida por su nombre seguido del conjunto de atributos, como se muestra a continuación:

NOME_RELACION (atributo 1, atributo 2, atributo 3,…,atributo n)

Por ejemplo, la relación *DIRECTOR* de una BD de venta de visionado de programas, podría definirse como:

DIRECTOR (idDirector, nome, apelido1, apelido2*, dataNacemento)

El hecho de que la relación se represente en forma de tabla es la causa de que los productos relacionales y los usuarios utilicen normalmente el nombre de tabla para referirse a las relaciones, y como consecuencia, se les llame filas a las tuplas y columnas o campos a los atributos. El grado de la relación es el número de columnas (atributos) de la tabla y la cardinalidad de la relación es el número de filas (tuplas) que tiene la relación en un momento dado. Por ejemplo, en la relación *DIRECTOR*, la representación en forma de tabla podría ser:



1.1.3.2 Dominio

Un dominio es un conjunto nombrado (con nombre) y finito de valores homogéneos (son todos del mismo tipo) y atómicos (son indivisibles). Por ejemplo, el atributo *género* en una relación *PROGRAMA* sólo podrá tomar los valores del dominio (drama, comedia, biografía, misterio, terror, romance, guerra, animación y aventuras). Cada dominio puede definirse de dos maneras:

- Mediante posibles valores.

  Por ejemplo, dominio de la clasificación de programa = todos los públicos, mayores de 9 años, mayores de 15 años, mayores de 18 años.

- Mediante un tipo de dato.

  Por ejemplo, dominio de la duración de un programa = valor de tipo numérico entero, o dominio del título = hasta 50 carácteres alfanuméricos.

A las veces se asocia una unidad de medida (kilos, metros, etc.) y/o ciertas restricciones (como un rango de valores).

El concepto de dominio compuesto aparece en ampliaciones del MR y se define como una combinación de dominios simples a la que pueden aplicase ciertas restricciones de integridad. Por ejemplo, el dominio compuesto denominado *Fecha* se construye por agregación de los dominios simples *Día*, *Mes* y *Año*, incorporando las adecuadas restricciones de integridad a fin de que no aparezcan valores no válidos para la fecha.

1.1.3.3 Atributo

Los atributos permiten definir las propiedades o características de la relación. Por ejemplo, la relación *DIRECTOR* dispondrá de los atributos *nombre* que representa el nombre propio del director o *fechaNacimiento* que contendrá la fecha en la que nació.

Un atributo y un dominio pueden llamarse igual, pero los atributos se diferencian de los dominios en los que:

- Un atributo está siempre asociado a una relación, mientras que un dominio tiene existencia propia con independencia de las relaciones.
- Un atributo representa una propiedad de una relación.
- Un atributo toma valores definidos en un dominio, estando cada atributo definido sobre un único dominio obligatoriamente.
- Un dominio puede contener valores no tomados por ningún atributo.
- Varios atributos distintos de la misma o de diferentes relaciones pueden tomar sus valores del mismo dominio.

La comparación de dos atributos sólo tendrá sentido si ambos toman valores del mismo dominio. Si el SGBD soporta la creación de dominios, podrá detectar este tipo de errores.

De la misma manera que es posible definir dominios compuestos, existen también atributos compuestos. Tanto los atributos compuestos como los dominios compuestos pueden ser tratados, si así lo precisa el usuario, como elementos únicos de información, es decir, como valores atómicos.

1.1.3.4 Valor nulo (NULL)

NULL no es un valor en sí mismo, sino que es un indicador o marca de ausencia de información y no debe confundirse con una cadena de caracteres vacía o con un valor numérico igual a cero. Se asocia el valor nulo (NULL) al valor desconocido de un atributo para una fila. El valor nulo puede permitirse o no en un atributo para reflejar situaciones del mundo real, como información perdida, ausencia de información o valores no aplicables a ese atributo.

Los atributos que admiten valores nulos se representa en la lista de atributos que siguen al nombre de la relación, con el nombre del atributo seguido de un asterisco.

1.1.3.5 Claves

Claves candidatas

Una clave candidata (CC, CANDIDATE KEY o CK) de una relación es un atributo o conjunto de atributos con valores únicos dentro de la relación, y por lo tanto, se puede usar para distinguir una tupla de todas las demás dentro de la relación. Por la propia definición de relación, siempre hay por lo menos una clave candidata, ya que en una relación no pueden existir tuplas repetidas. La clave candidata debe ser mínima, es decir, una parte de esa clave no puede ser su vez clave candidata.

Una relación puede tener más de una clave candidata. Se selecciona una de ellas como clave primaria (CP, PRIMARY KEY o PK), y las no seleccionadas serán claves alternativas o secundarias (CAI, ALTERNATIVE KEY o AK).

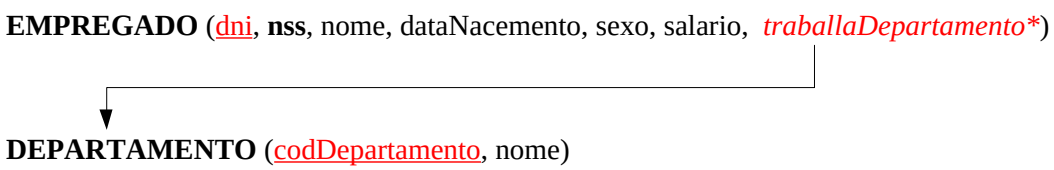
La representación de la clave primaria en la lista de atributos que siguen al nombre de la relación, se hace con el nombre del atributo en color rojo y subrayado, y la representación de las claves alternativas se hace con su nombre en negrita.

Claves ajenas o foráneas

Una clave ajena, o clave foránea (CA, FOREING KEY o FK), es un atributo o conjunto de atributos con valores que tienen que coincidir con los valores de la clave primaria de otra relación. Sirve para enlazar dos o más relaciones. La clave foránea y la correspondiente clave primaria han de estar definidas sobre los mismos dominios.

Por ejemplo, la relación *EMPLEADO*, posee el atributo *trabajaDepartamento* que indica el código del departamento en el que trabaja el empleado. La relación *DEPARTAMENTO* posee el atributo *codDepartamento* que indica el número del departamento y es la clave primaria. Se extiende que el atributo *trabajaDepartamento de* la relación *EMPLEADO es* una clave ajena o foránea, que hace referencia a la clave primaria *codDepartamento* de la relación *DEPARTAMENTO*.

La representación de la clave foránea en la lista de atributos que siguen al nombre de la relación, se hace con el nombre del atributo en color rojo y estilo de fuente cursiva, y desde ésta se dibujará una línea con punta de flecha que llegue hasta la relación a la que hace referencia. Por ejemplo:



1.1.4 Restricciones de integridad

Las restricciones de integridad permiten reflejar ciertas condiciones o reglas de negocio que debe cumplir el modelo de datos. Se pueden definir como estructuras u ocurrencias no permitidas en la base de datos, ya sean inherentes al propio modelo o definidas por el diseñador. Por ejemplo, en una BD de venta de visionado de programas, una restricción puede ser que cada película tenga un código asociado para distinguirlas de las series.

1.1.4.1 Restricciones inherentes o implícitas

Las restricciones inherentes se derivan de la misma estructura del modelo, no teniendo que ser definidas por el diseñador, ya que las impone el modelo. Pueden ser:

- Derivadas de la definición de relación:

  Cada relación tiene un nombre distinto dentro del conjunto de relaciones de un grafo.

  No hay dos atributos que se llamen igual dentro de la misma relación.

- Definidas por el modelo Relacional:
  - Restricción de clave. Toda relación tiene que tener a lo menos, una clave candidata, es decir, no existen tuplas repetidas, o dicho de otra manera, no puede haber dos filas con el mismo valor en todos los atributos.
  - Restricción de dominio. Cada atributo sólo puede tomar un único valor del dominio asociado. Por ejemplo, el atributo *telefono* en la relación *CLIENTE* sólo puede almacenar un número de teléfono para el cliente; si queremos tener almacenados varios teléfonos tendrían que definir otros atributos u otras relaciones donde guardarlos.
  - Restricción de integridad de entidad. Ningún atributo que forme parte de la clave primaria de una relación puede tomar un valor desconocido o inexistente, es decir, el valor nulo. Esto es porque el valor de la clave primaria sirve para identificar cada tupla y poder distinguir una tupla de todas las demás. Si dos o más tuplas tuvieran nulo en su clave primaria, no se podrían distinguir. Esta restricción debería aplicarse también a las claves alternativas, pero el modelo no lo exige.

1.1.4.2 Restricciones semánticas o de usuario

Las restricciones semánticas o de usuario son facilidades que el modelo ofrece a los diseñadores para que puedan reflejar la semántica del mundo real (Universo de Discurso) en el

esquema lo más fielmente posible. Son impuestas por el diseño en función de los requisitos del sistema a modelar, y por lo tanto, dependen del mismo. Los tipos de restricciones semánticas permitidos en el MR (incorporados en el estándar SQL 92), permiten aumentar su capacidad expresiva, y se detallan en los siguientes apartados.

Restricciones sobre atributos

- Restricciones sobre el dominio. Al definir cada atributo sobre un dominio, se impone una restricción sobre el conjunto de valores permitidos para cada atributo.
- Restricciones de obligatoriedad o de valor no nulo. Permiten declarar si uno o varios atributos deben tomar siempre un valor, es decir, no pueden tomar valores desconocidos o nulos.

Restrición de Clave Primaria (PRIMARY KEY)

Permite declarar un atributo o un conjunto de atributos como clave primaria de una relación. La clave primaria es la que el diseñador escoge de entre las claves candidatas, cumpliendo siempre la regla de integridad de entidad. Sus valores no podrán repetir ni se admitirán valores nulos (NULL). Hay que distinguir entre la restricción inherente de obligatoriedad de la clave primaria y la restricción semántica que permite al usuario indicar los atributos que forman parte de la clave primaria.

Aunque el modelo teórico impone esta restricción, ni SQL92 ni los SGBDR obligan a la declaración de una clave primaria para cada tabla, pero permiten la definición de la misma.

Por ejemplo, en el caso de la relación *EMPLEADO*, se escogió el atributo *dni* como clave primaria dejando el atributo *nss* (número de tarjeta de la seguridad social) como clave alternativa.

EMPREGADO (dni, **nss**, nome, dataNacemento, sexo, salario, *traballaDepartamento**)

Restricción de unicidad (UNIQUE)

Esta restricción establece que uno o varios atributos toman valores únicos, impidiendo que se repitan los valores que toman en diferentes tuplas. Normalmente se asocia a la definición de claves candidatas alternativas, aunque puede estar asociada a cualquier atributo.

Por ejemplo, en el caso de la relación *EMPLEADO*, las posibles claves candidatas serán el *dni* y el *nss*. En este caso se ha seleccionado el atributo *dni* como clave primaria y el atributo *nss* como clave alternativa o secundaria que debería tener asociada una restricción de unicidad.

Restricción de la integridad referencial (FOREIGN KEY)

Permite definir un atributo o un conjunto de atributos como clave ajena o foránea. La restricción permite enlazar relaciones o una relación consigo misma, a través de los valores de los atributos, de forma que el contenido de un atributo o conjunto de atributos en una relación (relación que referencia) debe coincidir con los valores de la clave primaria de la otra relación (relación referenciada). Los atributos que son clave foránea en una relación no precisan tener el mismo nombre que el atributo clave primaria con el que se corresponde.

La clave foránea puede ser simple (formada por un solo atributo) o compuesta (integrada por varios atributos). Cada componente de una clave foránea (FK) debe estar definido

sobre el mismo dominio que el correspondiente atributo de la clave primaria (PK) a la que hace referencia. El uso de claves foráneas facilitará la eliminación de las redundancias, y será el mecanismo del modelo relacional para establecer vínculos entre relaciones.

Por ejemplo, supongamos una base de datos relacional para un casting previo a un concurso de actores. El esquema está compuesto por varias relaciones que están enlazadas entre ellas mediante atributos definidos como claves ajenas. En la relación *ACTUACION* hay dos claves ajenas simples que son *codPapel* y *codActor* que hacen referencia a las relaciones *PAPEL* y *ACTOR* respectivamente. En la relación *PUNTUAR* hay una clave foránea simple que es *codJurado*, que hace referencia a la relación *MIEMBRO JURADO*, y una clave foránea compuesta por los atributos *codpapel* y *codActor* que hace referencia a la relación *ACTUACION*. La representación mediante tablas puede ser la siguiente:

ACTOR		
codActor	nome	departamento
456	Iria	18
678	Ximena	21
123	Breixo	32

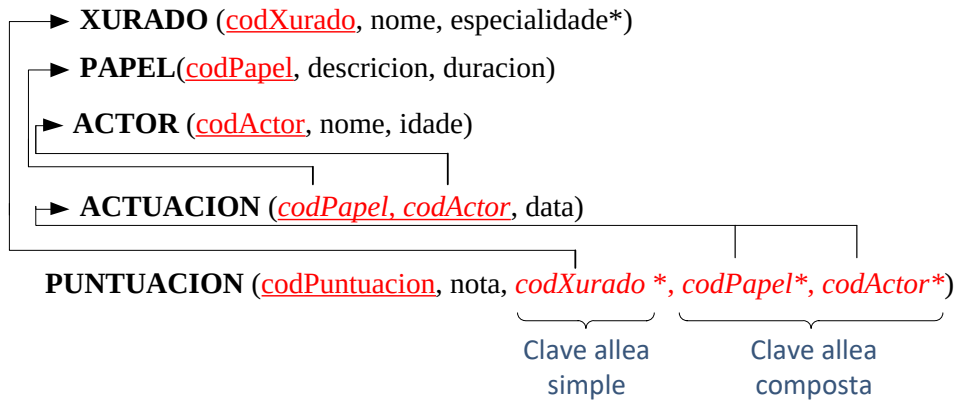
XURADO		
codXurado	nome	especialidade
222	Sabela	NULL
333	Roi	NULL
444	Xulia	NULL

PAPEL		
codPapel	descricion	duracion
1	rapaza	20
2	rapaz	17
3	malo	7
4	amiga	3

ACTUACION		
codPapel	codActor	data
1	456	03/03/2015
4	456	03/03/2015
1	678	04/03/2015
2	123	04/03/2015

PUNTUACION				
codPuntuacion	nota	codXurado	codPapel	codActor
1	3	222	1	678
2	5	333	4	456
3	7	333	2	456
4	6	444	2	123
5	8	222	1	678
6	6	NULL	3	678
7	5	333	NULL	NULL

El grafo relacional sería el siguiente:



NULOS y claves foráneas

El modelo relacional permite NULL como valor de una clave foránea. Por ejemplo, empleados no asignados a un departamento o empleados sin jefe. Teniendo en cuenta esto, podemos extender la definición de clave foránea indicando que es un atributo o conjunto de atributos de una relación (relación que referencia) que toman valores que han de coin-

cidir con el valor de la clave primaria de otra relación (relación referenciada), o que toman el valor nulo (NULL).

Por ejemplo, en la relación *PUNTUACION* anterior puede haber tuplas que contengan el valor nulo en la clave ajena simple *codJurado*. La tupla resaltada en azul en el siguiente gráfico es válida.

PUNTUACION				
codPuntuacion	nota	codXurado	codPapel	codActor
1	3	222	1	678
2	5	333	4	456
3	7	333	2	456
4	6	444	2	123
5	8	222	1	678
6	6	NULL	3	678
7	5	333	NULL	NULL

En el caso de claves foráneas compuestas, puede que todos los atributos que forman parte de la clave tomen el valor nulo (si estuviera permitido por el diseño), pero no puede ocurrir nunca que sólo alguno de los atributos que formen parte de la clave tomen el valor nulo. Por ejemplo, en la relación *PUNTUACION*, la tupla resaltada en rojo en el gráfico que se ve a continuación no es válida, porque *codPapel* forma parte de una clave foránea compuesta, y una clave foránea compuesta no puede tomar parcialmente el valor nulo. La tupla resaltada en azul es válida porque todos los atributos que forman parte de la clave foránea toman el valor nulo.

PUNTUACION				
codPuntuacion	nota	codXurado	codPapel	codActor
1	3	222	1	678
2	5	333	4	456
3	7	333	NULL	456
4	6	444	2	123
5	8	222	1	678
6	6	NULL	3	678
7	5	333	NULL	NULL

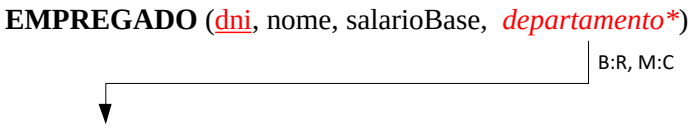
Políticas de mantenimiento de la integridad referencial

El establecimiento de claves foráneas obliga también a indicar las consecuencias que pueden tener las operaciones de borrado y modificación realizadas sobre las tuplas de la relación referenciada, para que se mantenga la integridad referencial después de hacer la operación. Estas consecuencias se representan en el grafo relacional con un texto junto a la línea que une la clave ajena con la relación en la que está la clave primaria a la que hace referencia. Los textos posibles son:

	Borrado (B)	Modificación (M)
Restringido (R)	B:R	M:R
En cascada (C)	B:C	M:C
Con puesta a nulos (N)	B:N	M:N
A un valor por defecto (D)	B:D	M:D

Para ilustrar los distintos casos posibles, se utilizará el siguiente grafo. La letra B hace referencia a la operación de borrado de tuplas en la relación *DEPARTAMENTO y* la letra M hace referencia a la operación de modificación de los valores de la clave primaria *codDe-*

partamento. Lo que va después de los dos puntos representa cómo debe reaccionar el SGBDR cuando se ordena una operación de borrado o modificación sobre la relación *DEPARTAMENTO*.



EMPREGADO			
dni	**nome**	**salarioBase**	**departamento**
33333333	José Arias Vázquez	800	1
11111111	María Paz Andrade	800	1
55555555	Antonio Rivas Zas	1000	3
66666666	Esther Lorenzo Arias	1200	4
77777777	Óscar Corredoira Fraga	1500	5
22222222	Jesús López García	1000	2
44444444	Elena Robles Becerra	1200	2

DEPARTAMENTO	
codDepartamento	**nome**
1	Vendas
2	Compras
3	Almacén
4	Contabilidade
5	Dirección

- Restringir (R = RESTRICT) las operaciones de borrado o modificación. Indica que no está permitido borrar o modificar la clave primaria en las tuplas de la relación que tiene la clave primaria referenciada, mientras existan tuplas en la relación que se referencia, en las que el valor de la clave foránea coincide con el valor de la clave primaria de la tupla que se quiere borrar o modificar.

  Por ejemplo, en el grafo relacional anterior, se indica mediante el texto B:R que no se pueden borrar tuplas en la relación *DEPARTAMENTO* mientras existan tuplas en la relación *EMPLEADO* en las que el valor de la clave ajena *departamento* coincida con el valor de la clave primaria de la tupla que se quiere borrar en la relación *DEPARTAMENTO*. Dicho de otra manera, no se podría borrar el departamento de código 5 en *DEPARTAMENTO* porque existe en *EMPLEADO* el empleado Óscar Corredoira Fraga que trabaja en ese departamento. En el caso de no restringir esta operación, podría haber tuplas de la relación *EMPLEADO* que hayan hecho referencia a departamentos que no existen en la base de datos, dando lugar a una inconsistencia de los datos.

- Aceptar y ejecutar las operaciones de borrado o modificación pero realizando acciones que restauren la integridad de los datos, pudiendo afectar a dos relaciones o más, si a su vez estas relaciones están unidas a otras, produciéndose cambios en cadena. Las acciones empleadas para restaurar la integridad referencial pueden ser:
  - Operación con transmisión en cascada (C = CASCADE). El borrado o la modificación de una tupla de la relación que contiene la clave primaria referenciada, obliga al borrado o la modificación en cascada de las tuplas de la relación que referencia, que tienen la clave foránea igual al valor de la clave primaria de la tupla afectada en la tabla referenciada.

    Por ejemplo, en el grafo relacional anterior, se indica mediante el texto M:C, que la modificación de la clave primaria de una tupla en la relación *DEPARTAMENTO*, obliga a la modificación del valor de la clave foránea (en cascada) en las tuplas en la relación *EMPLEADO* en las que el valor de la clave foránea *departamento* coincida con el valor de la clave primaria de la tupla que se quiere modificar en la relación *DEPARTAMENTO*. Dicho de otra manera, cuando se cambie en *DEPARTAMENTO* el código del departamento 1 por 8, se cambiarían automáticamente en *EMPLEADO* los códigos de los departamentos de José Arias Vázquez y María Paz Andrade, quedando las tablas como sigue:

EMPREGADO			
dni	**nome**	**salarioBase**	**departamento**
33333333	José Arias Vázquez	800	8
11111111	María Paz Andrade	800	8
55555555	Antonio Rivas Zas	1000	3
66666666	Esther Lorenzo Arias	1200	4
77777777	Óscar Corredoira Fraga	1500	5
22222222	Jesús López García	1000	2
44444444	Elena Robles Becerra	1200	2

DEPARTAMENTO	
codDepartamento	**nome**
8	Vendas
2	Compras
3	Almacén
4	Contabilidade
5	Dirección

– Operación con puesta a nulos (N = SET NULL). El borrado o la modificación de una tupla de la relación que contiene la clave primaria referenciada conlleva la puesta a nulos de los valores de la clave ajena de las tuplas de la relación que referencia, donde la clave ajena coincide con el valor de la clave primaria de la tupla afectada en la tabla referenciada.

Por ejemplo, si partiendo de la situación inicial de las relaciones, se coloca B:N en lugar de B:R en el grafo relacional, y si se borra en *DEPARTAMENTO* la tupla con el código de departamento 4, se provocaría que en *EMPLEADO* se pusiese a NULL el departamento de Esther Lorenzo Arias, quedando las tablas como sigue:

EMPREGADO			
dni	**nome**	**salarioBase**	**departamento**
33333333	José Arias Vázquez	800	1
11111111	María Paz Andrade	800	1
55555555	Antonio Rivas Zas	1000	3
66666666	Esther Lorenzo Arias	1200	NULL
77777777	Óscar Corredoira Fraga	1500	5
22222222	Jesús López García	1000	2
44444444	Elena Robles Becerra	1200	2

DEPARTAMENTO	
codDepartamento	**nome**
1	Vendas
2	Compras
3	Almacén
5	Dirección

– Operación con puesta a valor por defecto (SET DEFAULT). El borrado o la modificación de una tupla de la relación que contiene la clave primaria referenciada conlleva la puesta al valor definido como valor por defecto para los atributos que forman la clave foránea de las tuplas de la relación que referencia, donde la clave foránea coincide con el valor de la clave primaria de la tupla afectada en la tabla referenciada.

Por ejemplo, si la columna *departamento* de la tabla *EMPLEADO* se define en el momento de la creación de la tabla, con un valor por defecto igual a 5, y si partiendo de la situación inicial de las relaciones, se coloca B:D en lugar de B:R en el grafo relacional, en el caso de borrar en *DEPARTAMENTO* la tupla con el código de departamento 4, se provocaría que en *EMPLEADO* se pusies el valor por defecto para la columna *departamento* (tiene que ser un valor que exista en la tabla *DEPARTAMENTO*), quedando las tablas como sigue:

EMPREGADO			
dni	**nome**	**salarioBase**	**departamento**
33333333	José Arias Vázquez	800	1
11111111	María Paz Andrade	800	1
55555555	Antonio Rivas Zas	1000	3
66666666	Esther Lorenzo Arias	1200	5
77777777	Óscar Corredoira Fraga	1500	5
22222222	Jesús López García	1000	2
44444444	Elena Robles Becerra	1200	2

DEPARTAMENTO	
codDepartamento	**nome**
1	Vendas
2	Compras
3	Almacén
5	Dirección

Restricciones explícitas, basadas en el esquema o restricciones de negocio

Representan reglas de comportamiento del modelo que no pueden ser representadas en el grafo relacional, y en la fase de diseño lógico son redactadas en un documento aparte. Este grupo de restricciones son específicas de cada base de datos, y son definidas y almacenadas durante la fase de diseño físico en el esquema de la base de datos relacional mediante el uso del lenguaje SQL. Destacan las siguientes:

- Restriciones de verificación (CHECK). Permiten imponer condiciones a elementos o atributos de una relación. Se definirán con una sentencia CREATE TABLE y podrían tener nombre. Por ejemplo, la restricción de que el atributo *edad* de la relación *CLIENTE* tenga valores comprendidos entre 18 y 100 años, o que para la misma tupla, el atributo *fechaInicio tenga* valores menores que los del atributo *fechaFin*.
- Restricciones de aserción (ASSERTION). Son similares a las de verificación con la diferencia de que las condiciones pueden ser de atributos de más de una tabla. Por tanto, su definición no va unida a un determinado elemento del esquema y siempre ha de tener un nombre. Por ejemplo, la restricción de que un cliente tenga que ver como mínimo tres episodios de una serie para opinar sobre ella, o que los empleados que participan en un proyecto sean del mismo departamento que el responsable del proyecto.
- Restricción de disparadores (TRIGGERS). Restricción en la que el usuario puede especificar libremente la respuesta (acción) ante una determinada condición. Son objetos programados por el usuario para tal fin. Así como las anteriores reglas de integridad son declarativas, los disparadores son procedimentales, siendo preciso que el usuario escriba el procedimiento a aplicarse en el caso de que se cumpla la condición. Por ejemplo, hacer que cuando se da de alta una película, se incremente en una unidad el valor del atributo *numProgramasDirige* en la relación *DIRECTOR*, en la fila correspondiente al director de la nueva película.