

Introducción

Una vez que se ha obtenido una representación conceptual de las estructuras que almacenarán los datos, sus características y sus relaciones, es necesario transformarlas a otras que sean directamente implementables por las herramientas informáticas que se usarán en la siguiente fase de la implantación de una base de datos. Estas herramientas, que generalmente se les llama **sistemas gestores de bases de datos**, requieren de estructuras compatibles con ellas. Muchas de las soluciones actuales se basan en tablas o tuplas bidimensionales en las que se almacena la información. Este modelo se conoce como **modelo relacional** y está basado en lo que se conoce como **lógica de predicado** y la **teoría de conjunto**. Este modelo considera la base de datos como una colección de relaciones, donde estas se representan como tablas con una serie de filas donde se almacenan las instancias de un tipo de entidad o de un tipo de interrelación, y una serie de columnas provenientes de las propiedades de estos.

Una vez que se conoce si el sistema gestor de bases de datos que usar es relacional, se lleva a cabo la transformación de cada uno de los elementos representados en el diagrama conceptual, obteniendo así un grafo compuesto por todas las tuplas relacionadas. Este diagrama relacional equivale al esquema de una base de datos. El esquema contiene la definición de la estructura completa que permite almacenar los datos y sus relaciones.

■ 3.1. Del diseño conceptual al diseño lógico

En el diseño lógico se pretende construir y validar el modelo lógico de los datos. Esta fase posterior a la construcción de un modelo conceptual de datos tiene el objetivo de determinar las relaciones, validarlas tras procesos de normalización, analizar las restricciones de integridad repasando el modelo con el cliente y combinar los modelos que se obtienen en una visión global.

■■ 3.1.1. Elementos y notación del modelo relacional

Las relaciones representan en el diseño lógico lo que las entidades en el diseño conceptual. Una relación tiene un nombre, una serie de atributos y un conjunto de tuplas. El nombre debe ser único, es decir, no puede haber dos relaciones con el mismo nombre en la misma base de datos. Los atributos representan las propiedades inherentes de las relaciones y las tuplas, los valores que toman los diferentes atributos para cada elemento de la relación.

Las relaciones representan las tablas bidimensionales en las que se almacena la información. Las columnas de una tabla corresponden con los atributos de la relación y cada una de las tuplas son las filas de dicha tabla. Las tuplas coinciden con las ocurrencias de un tipo de entidad o de un tipo de interrelación y se llamarán **registros** cuando se transformen en tablas en el modelo físico.

Una relación no es lo mismo que una tabla en el modelo físico. El concepto relación hace alusión a un concepto más abstracto y está a medio camino entre un tipo de entidad o un tipo de interrelación y su materialización en tabla en el modelo físico.

En la Figura 3.1 se representa la relación llamada Asignatura que consta del código de la asignatura, CodAsig, su nombre, Nombre, el número de horas total que esta tiene asignada, NumHoras, y el código del ciclo formativo al que pertenece, CodCF. Cada una de estas filas representa una tupla de la relación.

ASIGNATURA			
CodAsig	Nombre	NumHoras	CodCF
1	Base de datos	165	1
2	Lenguaje de marcas	120	1
3	Seguridad informática	90	2
4	Despliegue aplicaciones web	110	1
5	Fundamentos de hardware	90	2
6	Acceso a datos	180	1

Figura 3.1. Representación de una tupla en la que se han introducido valores. También se puede llamar tabla.

En cuanto a los atributos, estos se pueden definir por extensión, por ejemplo, el código de la asignatura, CodAsig, tiene tres dígitos numéricos, o por extensión, especificando cada uno de los valores dentro del dominio.

3.1.2. Restricciones en el modelo relacional

Existe una serie de restricciones inherentes en los elementos del modelo relacional. Estas son:

- En una relación no puede haber dos tuplas iguales.
- El orden de las tuplas y el de los atributos no es relevante. La tabla de la Figura 3.2 es equivalente a la anterior.

ASIGNATURA			
CodAsig	CodCF	NumHoras	Nombre
1	1	165	Base de datos
2	1	120	Lenguaje de marcas
3	2	90	Seguridad informática
4	1	110	Despliegue aplicaciones web
5	2	90	Fundamentos de hardware
6	1	180	Acceso a datos


Figura 3.2. El orden de las columnas solo afecta en el momento de introducir los datos en las tablas, que se debe conocer en la medida de lo posible. Desde el punto de vista lógico esta tabla es análoga a la de la Figura 3.1.

- Cada atributo solo puede tomar un único valor del dominio, por ejemplo, Seguridad Informática no puede tener un 1 y un 2 en la columna CodCF.

- Ningún atributo que forme parte del atributo principal de una relación, clave primaria, puede tomar un valor nulo.

Los atributos pueden tener, además de las restricciones inherentes, otras de naturaleza semántica. Estas son la restricción de clave primaria, la de unicidad, la de obligatoriedad y la de clave ajena. A continuación, se detalla cada una de ellas:

- **Restricción de clave primaria o Primary key:** la restricción de clave primaria recae en el atributo o atributos que compongan la clave principal de una relación. Esta identifica a cada una de las tuplas posibles. Para indicar dicha restricción se usará la letra P delante del nombre del atributo.
- **Restricción de unicidad o Unique:** esta restricción recae en el atributo o atributos que compongan la clave alternativa. Se dice que es único porque los valores no pueden repetirse en diferentes tuplas, es decir, en dicha columna nunca puede haber dos valores iguales. Ejemplo: el número de la Seguridad Social, CodSS.
- **Restricción de obligatoriedad o Not Null:** esta restricción se usa para indicar que un atributo no puede tomar valores, es decir, no se puede quedar sin instanciar. No es lo mismo que haya una cadena vacía que un valor null. Un valor null indica que no se ha incluido nunca ningún valor. Si un atributo es opcional, entonces sí puede tomar el valor null.
- **Restricción de clave ajena o Foreign key:** esta restricción indica que un atributo o atributos conforman la clave primaria en otra tabla y se usa para relacionar una tabla con otra. Es lo que se conoce como integridad referencial y no tiene por qué tener el mismo nombre que la clave ajena. En la tabla anterior llamada asignatura, el atributo CodCF comportaría la clave ajena y apuntaría a la clave primaria de la tabla Ciclo-Formativo (de tal forma que las tablas quedan relacionada a través de su clave ajena).



ASIGNATURA			
CodAsig	Nombre	Horas/semana	CodCF
1	Basics de datos	200	1
2	Lenguaje de marcado	120	2
3	Seguridad informática	80	3
4	Desarrollo aplicaciones web	110	4
5	Fundamentos de hardware	80	5
6	Acceso a datos	100	6

CICLO	
CodCF	Nombre
1	Desarrollo de aplicaciones web
2	Administración de sistemas informáticos en red
3	Desarrollo de aplicaciones multiplataforma

Figura 3.3. En la tabla de la izquierda hay una columna, CodCF, con la restricción de clave ajena. Esta deriva de la columna CodCF con restricción clave primaria de la tabla de la derecha.

Es importante advertir que en la columna CodCF de la tabla de la izquierda, Asignatura, nunca habrá un valor que no exista en la columna CodCF de la tabla de la derecha, Ciclo. En el caso de un intento de insertar un valor que no existe en esta última, dicha inserción será invalidada por no cumplir la restricción de clave ajena.

Ten en cuenta

Clave ajena o clave foránea, en inglés, se escribe *Foreign key*. Es un error muy habitual escribir *Foreing* en vez de *Foreign*.





Figura 3.4. La elección de las claves primarias y su propagación en claves foráneas son las bases de un buen diseño en la planificación de las bases de datos relacionales.

3.1.3. Notación en el diseño lógico

En la etapa de diseño lógico, se obtiene un grafo compuesto por un conjunto de relaciones que interactúan entre ellas a través de sus claves ajenas. Dicho diagrama lógico se representa a través de dichas tuplas compuestas por el nombre de la relación y el conjunto de atributos encerrados entre paréntesis y separados por coma. A cada atributo se le precederá de letras de las restricciones particulares que posean. Se usarán diferentes letras para representar cada restricción.

- **Restricción de clave primaria:** se usa la letra P, igual que se usaba en el diseño conceptual y equivale también a $\text{—}\bullet$. Se le coloca la letra P a cada atributo que forma la clave primaria. Ejemplo:

Profesor(P-dni, P-letra, nombre, PrApellido, SgApellido)
CicloFormativo(P-CodCF, Siglas, nombre, tipo)

- **Restricción de clave secundaria:** se usa la letra U, igual que se usaba en el diseño conceptual y equivale también a $\text{—}\circ$. Se coloca la letra U a cada atributo que forma la clave secundaria. Por ejemplo, para las tablas Profesor y CicloFormativos, se indica haciendo uso de la letra U, que los atributos número de la Seguridad Social y siglas son únicos.

Profesor(P-dni, P-letra, U-NumSegSoc, nombre, PrApellido, SgApellido)
CicloFormativo(P-CodCF, U-Siglas, nombre, tipo)

- **Restricción de obligatoriedad:** se usa la letra N para indicar que el atributo puede tomar nulos (NULL), al igual que se usaba en el diseño conceptual y equivale también a $\text{—}\circ$. Ejemplo: el segundo apellido es opcional y, por lo tanto, puede tomar el valor null.

Profesor(P-dni, P-letra, U-NumSegSoc, nombre, PrApellido, N-SgApellido)



Se usa el símbolo N para indicar que puede tomar Null; no confundir con el valor Not Null, que, como suele aplicarse en casi todos los atributos de una relación, en su representación relacional, no se usa ningún símbolo para indicarlo.

- **Restricción de claves ajenas:** las claves ajenas apuntarán a las claves primarias de otras relaciones y para ello se usará una flecha después del nombre del atributo e indicando la relación a la que se refiere. La letra que se usará para indicar que un atributo comporta una clave ajena es F (Foreign). Las claves foráneas no se indicaban en el diseño conceptual. Siguiendo con el ejemplo de las asignaturas y de los ciclos formativos, las dos relaciones que se obtienen tras añadir la clave ajena a Asignatura que apunta a CicloFormativo son:

Profesor(P-dni, P-letra, U-NumSegSoc, nombre, PrApellido, N-SgApellido,
F-CodCF → **CicloFormativo**)

CicloFormativo(P-CodCF, U-Siglas, nombre, tipo)

En una clave foránea le precede la letra F a todos los atributos que la forman, y se indica luego la relación a la que apunta. Por ejemplo, si se añade la relación Asignatura y se decide que un profesor puede impartir muchas asignaturas, pero una asignatura solo puede impartirla un profesor, los cambios del grafo relacional quedan como sigue:

Profesor(P-dni, P-letra, U-NumSegSoc, nombre, PrApellido, N-SgApellido,
F-CodCF → **CicloFormativo**)

CicloFormativo(P-CodCF, U-Siglas, nombre, tipo)

Asignatura(P-CodAsig, siglas, nombre, numHoras, F-dni, F-letra → **Profesor**)

Evidentemente una relación puede tener muchas claves ajenas, cada una de ellas apuntando a la clave primaria de la relación de la que depende. Es sencillo separar todas las claves ajenas en la notación usada, ya que cada atributo o atributos que la forman se separa cuando se indica a la relación a la que apunta. Por ejemplo, si se desea registrar el departamento al que pertenece un profesor, a la relación Profesor se le añadiría la clave primaria de la relación departamento. La solución queda como se ve en la Figura 3.5.

Departamento (P-CodDep, nombre, ubicacion)
CicloFormativo(P-CodCF, U-Siglas, nombre, tipo)
Profesor(P-dni, P-letra, U-NumSegSoc, nombre, PrApellido, N-SgApellido, F-CodCF → **CicloFormativo**,
F-CodDep → **Departamento**)
Asignatura(P-CodAsig, siglas, nombre, numHoras, F-dni, F-letra → **Profesor**)

Figura 3.5. En la relación Asignatura existe una clave ajena compuesta por dos atributos dni y letra. Se identifican claramente porque se agrupan antes de indicar con la flecha a qué tabla apunta.

Se puede observar que en la relación Profesor hay dos claves ajenas, una que apunta a la relación CicloFormativo y otra que lo hace a Departamento, y que es fácil distinguir los atributos que comportan cada clave ajena.

3.2. Transformación de los tipos de entidad y los atributos

Una vez que se ha presentado las consideraciones generales en la transformación de esquemas conceptuales a relacionales, es necesario estudiar cada una de las reglas de transformación entre los diferentes elementos del diseño conceptual, para obtener un conjunto de tuplas relacionadas con eficiencia. Estas reglas comportan un conjunto de heurísticas que mecanizan el sistema de transformación. Las transformaciones son tan mecánicas que existen muchos programas comerciales que hacen esta tarea. Por ejemplo, con el programa DataModeler de Oracle se pueden obtener diagramas lógicos totalmente optimizados partiendo del diagrama conceptual que diseña el usuario de la aplicación.

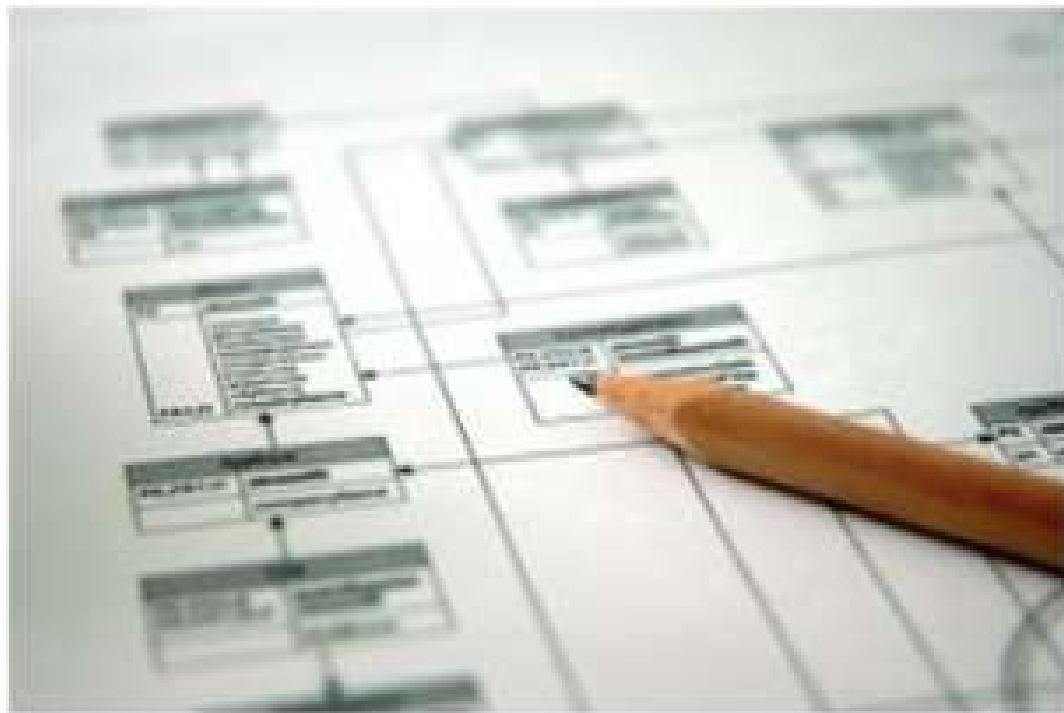


Figura 3.6. El esquema relacional estará compuesto por relaciones (futuras tablas) que se relacionan con otras a través de sus claves primarias (PK) y ajenas (FK).

Cada atributo de una entidad se transforma en una columna de una tabla. Los atributos principales pasan a formar parte de la clave primaria de la tabla. En el diagrama relacional se representa con una P delante de cada uno de esos atributos que comporta dicha clave primaria. Los atributos secundarios, o también llamados **alternativos**, serán la clave secundaria y tiene su restricción Unique. Se representa con una U.

A los atributos obligatorios se les añade la restricción NOT NULL. No se utiliza ninguna letra para indicar esto, ya que la mayoría de los atributos son obligatorios. Los atributos opcionales son aquellos que pueden tomar valores nulos en su dominio. Se usará la letra N para indicar que son opcionales.

Los atributos multivaluados se convierten en una nueva tabla cuyos únicos atributos serán la concatenación de la clave primaria del tipo de entidad y el atributo multivaluado. Además, se creará una clave ajena que hace referencia a la tabla principal.

Los atributos derivados se indican con una D, y son atributos con un cálculo asociado a cada ocurrencia. Para los atributos compuestos se crea una columna para cada uno de los atributos que los componen.

Los atributos codificables (representados con la letra C) se convierten en una tabla. El nombre de la tabla será el nombre del atributo, con campo clave una secuencia y un campo descripción. También se puede usar directamente la descripción como campo clave. La tabla que posea el atributo codificable hereda como clave ajena la clave primaria de esta tabla recién creada apuntando a ella.

ALUMNO	ALUMNO	ALUMNO
P - dni U - NSS Nombre Ap1 N - Ap2 MN - Telefono C - Localidad	P - dni (L,1) U - NSS (L,1) Nombre (L,1) Ap1 (L,1) N - Ap2 (O,1) MN - Telefono (D,n) C - Localidad (L,1)	→ dni → NSS → Nombre → Ap1 → Ap2 → Telefono → Localidad

Figura 3.7. Se observan diferentes maneras de representar las restricciones de los atributos de un tipo de entidad.

En la Figura 3.7 se indican las tres maneras diferentes que se han usado para representar un tipo de entidad llamado *Alumno*, y sus diferentes atributos atendiendo a diferentes tipos: Principal o primario (P), Secundaria o Único (U), opcional (N), multivaluado (M) y codificable (C). A continuación, se indica en la figura la transformación de los atributos de dicho tipo de entidad.

Localidad(P-CP, descripción)
Alumno(P-dni, nombre, U-NSS, op1, N-op2, F-CP → Localidad)
Telefono (P-NumTno, F-dni → Alumno)

Figura 3.8. Transformación al esquema relacional de atributos simples, únicos, U, multivaluados, M, y codificables, C. El atributo multivaluado Telefono se ha convertido en una nueva tabla. En la tabla Telefono se usa dni como una clave foránea que apunta a la tabla Alumno.

La transformación del dominio consiste en declarar una lista por Intensión o por extensión de los valores que puede tomar un campo. Por ejemplo, CHECK(UPPER(TipoLocalidad IN ('Aldea','Pueblo','Ciudad','Capital'))).

Actividad propuesta 3.1.

Transformación de entidades y atributos a tuplas y columnas

Obtén el esquema relacional de los tipos de entidad que se representan en las Figuras 2.5, 2.7 y 2.8.

3.3. Transformación de las interrelaciones

Las reglas de transformación de las interrelaciones 1:1, 1:N y N:M son análogas, sean binarias, binarias reflexivas o ternarias. Sin embargo, atendiendo a consideraciones

semánticas o dependencias con las cardinalidades mínimas y máximas, estas reglas pueden incluir diferentes consideraciones para su transformación. A continuación, se estudian estas consideraciones para las relaciones 1:1, 1:N, N:M tanto en binarias como en reflexivas y ternarias.

3.3.1. Transformación de las interrelaciones 1:1 y 1:N

Existen dos maneras de transformar las relaciones 1:N. Primero, propagando el campo clave del tipo de entidad que tiene la cardinalidad más baja a la que tiene la cardinalidad más alta. Cuando el número de ocurrencias interrelacionadas del tipo de entidad que propaga su clave es muy pequeño, se transforma la interrelación en una tabla como si se tratara de una interrelación N:M, pero ahora la clave primaria de la tabla creada es solo la clave primaria de la tabla a la que le corresponde la cardinalidad mayor. Esto también se hace cuando se prevé que la relación se puede convertir en una tabla N:M o cuando la interrelación tiene atributos propios y no es deseable propagarlos para conservar la semántica.



Figura 3.9. Las claves foráneas, *foreign key*, son el elemento fundamental de las bases de datos relacionales, ya que son las que comportan las relaciones entre las tablas.

Con el requisito «Se desea registrar las asignaturas que imparte cada docente, sabiendo que una asignatura no puede estar impartida por más de un profesor», se obtienen dos tipos de entidad llamadas Asignatura y Profesor, y una relación Imparte con multiplicidad 1:N. Supóngase los atributos y las cardinalidades mínimas y máximas que se indican en la Figura 3.10.

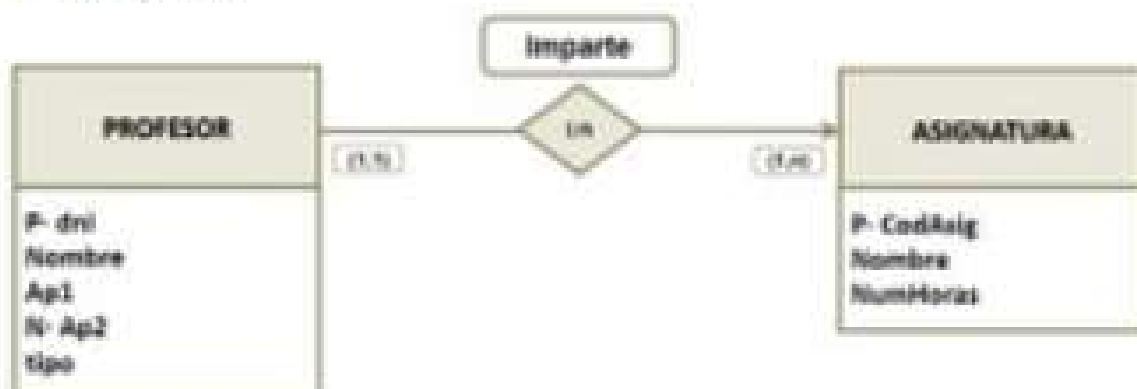


Figura 3.10. Interrelación 1:N entre Profesor y Asignatura para registrar cada una de las asignaturas que imparte un profesor. El sentido es 1:N desde Profesor a Asignatura.

Usando la técnica de propagación, el campo clave de Profesor, dni, se añade como clave ajena a Asignatura. Como la cardinalidad mínima es uno, la clave ajena es obligatoria. El resultado en la representación relacional se recoge en la Figura 3.11.

```
Profesor(P-dni, Nombre, Ap1, N-Ap2, tipo)
Asignatura(P-CodAsig, Nombre, NumHoras, F-dni → Profesor)
```

Figura 3.11. La interrelación 1:N de la Figura 3.10 se ha convertido en una clave ajena en la tabla Asignatura denominada dni, que apunta a la tabla Profesor, es decir, el dni que se incluya en dicho lugar es el campo clave del profesor incluido en la tabla Profesor y es el profesor que imparte dicha asignatura.

Es importante observar que, cuando se da una ocurrencia del tipo de entidad Asignatura, es obligatorio indicar un valor en la columna dni que apunta al profesor que imparte dicha asignatura. Si el profesor no existe o no se sabe aún quién lo va a impartir, no se puede registrar dicha ocurrencia. Será necesario estudiar si se desea que la integridad referencial esté por encima del almacenamiento de la ocurrencia. En tal caso, la cardinalidades en Profesor sería (0,1) en vez de (1,1), es decir, una asignatura puede no tener un profesor, admitiendo valores nulos. Esto puede provocar falta de integridad referencial, pero permite dar de alta a una asignatura cuando no se conoce el profesor con el que se relaciona. Evidentemente, es más importante evitar la falta de integridad referencial, pero, a veces, es inevitable. En tal caso hubiera quedado tal y como se recoge en la Figura 3.12.

```
Profesor(P-dni, Nombre, Ap1, N-Ap2, tipo)
Asignatura(P-CodAsig, Nombre, NumHoras, FN-dni → Profesor)
```

Figura 3.12. Se puede considerar una clave ajena como opcional, cuando al dar de alta a una asignatura no se conoce aún el profesor que la impartirá. En tal caso, para que la instancia quede registrada en Asignatura, la columna dni debe admitir valores nulos.

Cuando la interrelación es de tipo 1:1, se pueden aplicar las opciones antes expresadas. Generalmente se propaga la clave del lado que interese al otro. A veces, si un campo clave puede tomar nulos, interesa propagar con el sentido que menos dificulte el almacenamiento de la ocurrencia. Estas consideraciones se basan en las cardinalidades mínimas, para recoger la mayor semántica posible, evitar valores nulos o para aumentar la eficiencia.

Sin embargo, si las cardinalidades mínima y máxima en ambos tipos de entidad son (0,1), es necesario crear una tabla. Si las entidades que participan en la interrelación poseen cardinalidades (0,1) y (1,1) respectivamente, conviene propagar la clave del tipo de entidad con cardinalidades (1,1) a la tabla resultante del tipo de entidad con cardinalidades (0,1).

Con el requisito «Se desea registrar el profesor que es jefe del departamento al que pertenece, sabiendo que un profesor solo pertenece a un departamento, y que, obviamente, un departamento está compuesto por un conjunto de profesores que pertenecen a él», y considerando que Profesor y Departamento son dos tipos de entidad, se obtiene una relación 1:1 entre profesor y departamento para el almacenamiento de su jefatura. El diseño conceptual podría ser el de la Figura 3.13.

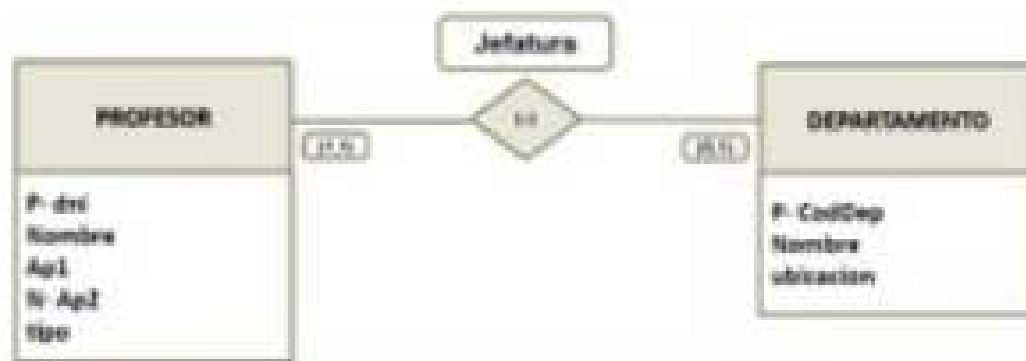


Figura 3.13. En esta figura se representa la relación de Jefatura con multiplicidad 1:1 entre el tipo de entidad Profesor y el tipo de entidad Departamento. Un departamento solo tiene como jefe un profesor, y el que lo sea solo lo será de un departamento. La cardinalidad mínima en el sentido Profesor-Departamento es cero, ya que no todo profesor es jefe de un departamento.

Se considera las cardinalidades (0,1) en el lado de profesor, pues el profesor puede ser jefe como mínimo cero, nunca, y como máximo de un departamento, y (1,1) en el lado del departamento, es decir, todo departamento tiene como mínimo un profesor que es jefe y como máximo uno también. Al ser una relación con multiplicidad 1:1, se podría propagar la clave primaria de un lado al otro indiscriminadamente. Pero en este caso, interesa propagar el dni del profesor como clave ajena en la tabla Departamento, ya que todo departamento va a tener un dni en dicha columna. Si se hubiera hecho al revés, aquellos profesores que no son jefes tendrían un null como valor. Si se prevé que una clave ajena va a contener muchos valores nulos, interesa propagarla hacia el otro sentido de la relación. El resultado en la representación relacional sería el que se observa en la Figura 3.14.

Profesor(P-dni, Nombre, Ap1, N-Ap2, tipo)
Departamento(P-CodDep, Nombre, Ubicación, F-dni → Profesor)

Figura 3.14. La transformación 1:1 es como la 1:N. Cuando ambos sentidos tienen la misma cardinalidad mínima se puede hacer en un sentido o en el otro. En el caso de la Figura 3.13, el sentido que tiene menor cardinalidad es de Profesor a Departamento.

Nota técnica



La transformación de una relación 1:1 y 1:N genera un nuevo campo en una de las dos tablas que une, siendo este campo el que las relaciona. Este campo, llamado clave ajena, debe tener el mismo dominio en ambas tablas. Cuando un tipo de entidad posee más de un atributo en su campo clave o identificativo, la clave ajena también está compuesta por el mismo número de campos. Estos campos, los de la clave primaria y los de la clave ajena, no tienen por qué llamarse iguales, pero es ventajoso que así se haga para facilitar posteriores acciones. Es muy importante estudiar la conveniencia de añadir la clave ajena en una tabla que en la otra. Además, decidir si esta clave ajena aceptará nulos es bastante importante, ya que, si se exige que sea obligatoria, es decir, es obligatorio insertar un valor en dicha columna para una ocurrencia del tipo de entidad a la que se le añade la clave ajena, se puede incurrir en la obligatoriedad de insertar un registro con dicho valor ajeno y que aún no exista en la tabla que se va a relacionar.

Actividad propuesta 3.2.

Transformación relaciones 1:1 y 1:N

En el esquema conceptual de la Figura 3.15 se representan dos interrelaciones entre el tipo de entidad Profesor y el tipo de entidad Departamento. La primera es para registrar cada uno de los profesores que compone cada departamento. La segunda es para registrar el profesor que es jefe de departamento. Transforma el diseño conceptual a su esquema relacional.

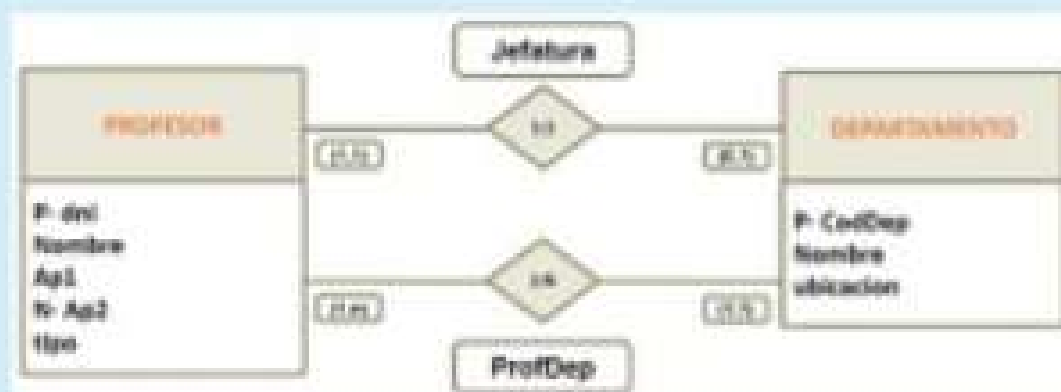


Figura 3.15. Entre Profesor y Departamento de este ejemplo existen dos relaciones, una 1:1 para registrar el jefe de cada departamento, y otro 1:N para registrar los profesores de cada departamento.

3.3.2. Transformación de las interrelaciones N:M

Un tipo de interrelación N:M se transforma en una tabla que tendrá como clave primaria la concatenación de las claves primarias de los dos tipos de entidad que relaciona. Además, esas claves son ajenas y referencian a cada tabla de la que proviene. Si la interrelación posee atributos normales, estos serán añadidos a la tabla nueva. Si algún atributo connota aspectos temporales o históricos debe considerarse como participe en la clave principal de la tabla recién creada.

Con el requisito «Se desea registrar las asignaturas en las que se matricula cada alumno. Se hace necesario registrar su nota final», se obtiene dos tipos de entidad llamadas asignatura y alumno, y una interrelación entre ellas, *Inscripción*, con multiplicidad N:M y con el atributo calificación. El resultado conceptual sería el de la Figura 3.16.

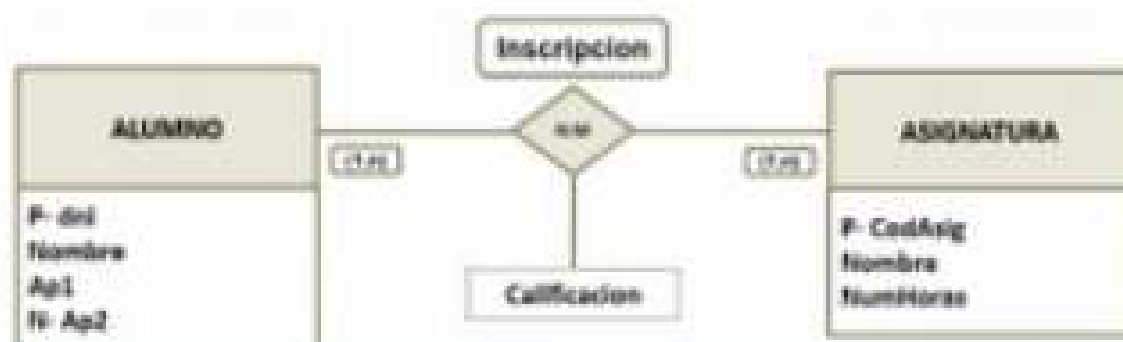


Figura 3.16. Cuando un alumno se relaciona con una asignatura aparece un nuevo atributo que es la calificación que este obtiene. La interrelación es N:M, ya que un alumno puede inscribirse en muchas asignaturas y en una asignatura lo pueden hacer muchos alumnos.

Como la multiplicidad es N:M, se genera una tabla nueva llamada *Inscripcion*, cuyo campo clave está compuesto por los campos claves de los dos tipos de entidad que une, es decir, *dni* y *CodAsig*, y, a su vez, son claves ajenas que apuntan cada una a la tabla de la que viene. El atributo de la relación llamado *Calificación* se le añade a esta tabla. El resultado sería como se recoge en la Figura 3.17. En este caso, se podría considerar que el atributo *Calificación* acepta nulos, ya que cuando el alumno se inscribe en una asignatura aún no se conoce la calificación obtenida.

```
Alumno(P-dni, Nombre, Ap1, N-Ap2)
Asignatura(P-CodAsig, Nombre, NumHoras)
Inscripcion(PF-dni → Alumno, PF-CodAsig → Asignatura, Calificación)
```

Figura 3.17. Toda relación N:M se convierte en una nueva tabla con campo clave los dos de las entidades que unen y como clave ajena a cada tabla que apunta. Los atributos de una interrelación se propagan junto a su clave ajena.

Si el requisito anterior hubiera exigido que el alumno se pueda matricular más de una vez, es necesario añadir a la relación el atributo temporal *fecha*. Este atributo formará parte de la clave primaria de la interrelación *Inscripcion*.

```
Alumno(P-dni, Nombre, Ap1, N-Ap2)
Asignatura(P-CodAsig, Nombre, NumHoras)
Inscripcion(PF-dni → Alumno, PF-CodAsig → Asignatura, P-fecha, Calificación)
```

Figura 3.18. Cuando se quiere que la relación entre dos ocurrencias pueda darse más veces a lo largo del ciclo de vida de la base de datos, se debe incluir un atributo *fecha* como parte de la clave primaria de la tabla nueva.

Su representación conceptual hubiera sido como se indica en la Figura 3.19.

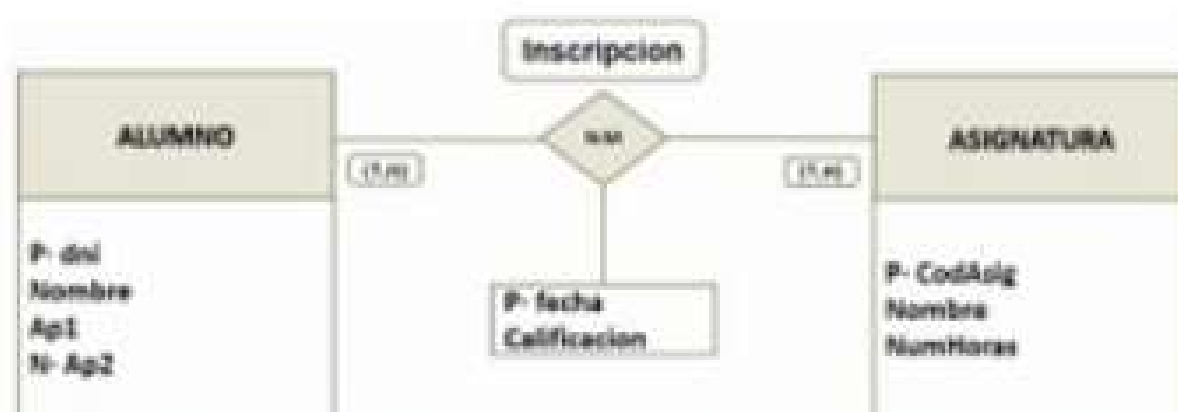


Figura 3.19. El atributo *fecha* de la interrelación *Inscripcion* tiene la letra *P* para indicar que esta fecha formará parte de la clave primaria con el fin de que las mismas ocurrencias puedan relacionarse más veces.

Actividad propuesta 3.3.

Transformación de interrelaciones al esquema relacional

Obtén el esquema relacional del diagrama representado en la Figura 2.17.