Apuntes accesibles UT3-Relacional

Sitio: FRANCISCO DE GOYA
Curso: Bases de Datos Pendientes
Libro: Apuntes accesibles UT3-Relacional

Imprimido por: Lucía Hernández Montero
Día: lunes, 24 de octubre de 2022, 09:33

Tabla de contenidos

1. El modelo Relacional

El objetivo principal del modelo relacional es proteger al usuario de la obligación de conocer las estructuras de datos físicas con las que se representa la información de una BBDD, así se permite que la BBDD pueda implementarse en cualquier gestor de BBDD relacional (SQL). Las características de este modelo es:

- La relación es el elemento fundamental del modelo. Estas relaciones se pueden operar mediante el Álgebra Relacional.
- El modelo relacional es independiente de la forma en que se almacenan los datos y de la forma de representarlos, por tanto la BBDD se puede representar en cualquier SGBD.
- Al estar fundamentado en una fuerte base matemática, se puede demostrar la eficacia del modelo a la hora de operar conjuntos de datos.

1.1. Las relaciones del modelo Relacional

Se define una relación como un conjunto de atributos, cada uno de los cuales pertenece a un dominio, y que posee un nombre que identifica la relación. Se representa gráficamente por una tabla con columnas (atributos) y filas (tuplas). El conjunto de tuplas de una relación representa el cuerpo de la relación y el conjunto de atributos y el nombre representan el esquema.

Actualmente los modelos lógicos más extendidos con diferencia son el modelo relacional y los diagramas de clases que utiliza UML para modelar las BBDD orientadas a objetos. El modelo relacional de Cood se ajusta a la perfección al modelo E/R de Chen.

Relación Alumnos

Matrícula	Nombre	Apellidos	Curso	Nota
3256	José	Pérez de Lastra	1	5,25
0101	María	Anúnez Sastre	2	7,8
8743	Lourdes	Sánchez Argote	1	4,5
1234	Antonio	Soria Madrid	3	6,35
5674	Luís	González Silos	1	3,25
0678	Pilar	Alcántara Badajoz	2	5,5
0346	Dolores	Almiz Márquez	3	7,3
2985	Manuel	Rivas Fuentes	3	3,5

Una **tupla** es un conjunto coherente de datos, que representan una instancia de una entidad, por lo que en la tabla Alumno una tupla puede ser (3256, José, Pérez Lastra, 1, 5,25). De esta forma la tabla alumnos tiene 8 tuplas.

Los **atributos**serán las columnas de la tabla. La tabla tendrá un **nombre**. El **esquema** es el conjunto de atributos junto con el nombre de la tabla, que será con lo que se trabajará a lo largo de la primera mitad de la unidad de trabajo. El **cuerpo**de la tabla se corresponde con el conjunto de todas las tuplas y finalmente, el **estado** está formado por el esquema y el cuerpo.

Un esquema de una relación se representará de la siguiente forma, tomando por ejemplo la tabla de Alumno:

Alumno (matrícula, nombre, apellidos, curso, nota).

Esta relación no representa las restricciones que se verán posteriormente y se añadirán a esta forma de escribir el esquema.

1.2. Características del modelo

Vamos a definir los conceptos necesarios para transformar el modelo conceptual (diagrama E/R) en el modelo lógico (modelo relacional).

- **Atributos**: características que describen a una entidad o relación.
- **Dominio**: conjunto de valores permitidos para un atributo. Por ejemplo, cadena de caracteres, número enteros, los valores Sí o No, etc.
- **Restricciones de semántica**: condiciones que deben cumplir los datos para su correcto almacenamiento. Hay varios tipos:
  - Restricciones de clave: es el conjunto de atributos que identifican de forma única a una entidad.
  - Restricciones de valor único (UNIQUE): impide que un atributo tenga un valor repetido. Todos los atributos clave cumplen esta restricción. Puede ser que algún atributo que no sea clave sea necesaria esta restricción, por ejemplo, el número de bastidor de un coche, que no es clave principal, lo es matrícula, no se puede repetir.
  - Restricciones de integridad referencial: se da cuando una tabla tiene una referencia a algún valor de otra tabla. En este caso la restricción exige que exista el valor referenciado a la otra tabla. Por ejemplo, no se puede poner una nota a un alumno que no exista.
  - Restricciones de dominio: exige que el valor que puede tomar un campo este dentro del dominio definido. Por ejemplo, si se establece que un campo DNI pertenece al dominio de los números de 9 dígitos + 1 letra no es posible insertar un DNI sin letra.
  - Restricciones de verificación (CHECK): permite comprobar si un valor de un atributo es válido conforme a una expresión.
  - Restricciones de valor nulo (NULL o NOT NULL): un atributo puede ser obligatorio si no admite el valor NULO o NULL. Si admite como valor el valor NULL el atributo es opcional.
  - Disparadores o triggers: son procedimientos que se ejecutan para hacer una tarea concreta en el momento de insertar, modificar o eliminar información de un tabla.
  - Restricciones genéricas adicionales o aserciones (ASSERT): permite validar cualquiera de los atributos de una o varias tablas.
- **Clave**: una clave es un conjunto de atributos que identifican de forma única una ocurrencia de entidad, por lo que implementan ciertas restricciones. Las claves pueden ser simples (atómicas) o compuestas. Hay varios tipos de claves:
  - Superclave o **clave primaria (PK)**: identifican a una entidad. Por ejemplo, para un empleado, las superclaves posibles son el DNI, o el DNI+Nombre o el DNI+Nombre+Número de la Seguridad Social, etc. Para indicarlo de forma textual, o accesible, se escribe (PK) a la derecha del atributo si es solo uno, y si es compuesto por dos o más atributos se pondrá después del último atributo (PK atributo1, atributo2...) antes de cerrar la relación. De forma gráfica se subrayará.
  - Clave candidata: La clave candidata de una relación es el conjunto de atributos que determinan unívoca y mínimamente cada tupla. Siempre hay una clave candidata pues por definición no puede haber dos tuplas iguales En una relación pueden existir varias claves candidatas.
  - **Clave ajena (FK)**: es un atributo de una entidad que es la clave primaria de otra entidad. Para representarlo textualmente (accesible) se indicará a la derecha del atributo con (FK:relación) donde la relación es la relación referenciada. De forma gráfica se realizará un subrayado discontinuo y mediante una flecha se apuntará a la relación referenciada.

Ejemplo:

Empleados: (código (PK), dni, dirección, telefono, sueldo, comisión, cod_dep (FK:Departamentos)).

Departamentos: (cod_dep (PK), nombre, localidad)

En la relación Empleados podrían ser claves candidatas tanto código como DNI y la conjunción de ambas. El diseñador elige cuál de ellas será la clave primaria que en este caso será código. Igualmente en departamento cod_dep será la clave primaria. Por otro lado, cod_dep en Empleado es una clave ajena (FK) de Departamentos. Por ello, toda tupla Empleado que tenga un valor en cod_dep, ese mismo valor ha de existir en Departamentos.

Otro ejemplo:

Editorial: (cod_edit (PK), nombre, dirección, ciudad)

Libro: (codigo (PK), titulo, idioma, cod_edit (FK: Editorial))

Escribe: (cod_autor (FK:Autor), cod_libro (FK:libro), (PK cod_autor, cod_libro)).

Autor: (cod_autor(PK), nombre, nacionalidad)

Donde cod_edit en Libro es clave ajena que referencia a cod_edit de Editorial. cod_libro en Escribe es clave ajena de Libro y cod_autor es clave ajena de Autor.

1.3. Integridad referencial

La integridad referencial es implementada por las claves ajenas, tal y como se ha indicado anteriormente. La regla indica que "*los valores de la clave ajena o bien coinciden con los de la clave primaria a la que referencian o bien son nulos*". A diferencia de la clave primaria, una clave ajena puede ser nula o si el diseñador así lo elige, impedir que sea nula. Pero para poder mantenerse íntegra ha de indicarse qué es lo que sucede cuando un valor que es referenciado desde otras relaciones es modificado o eliminado.

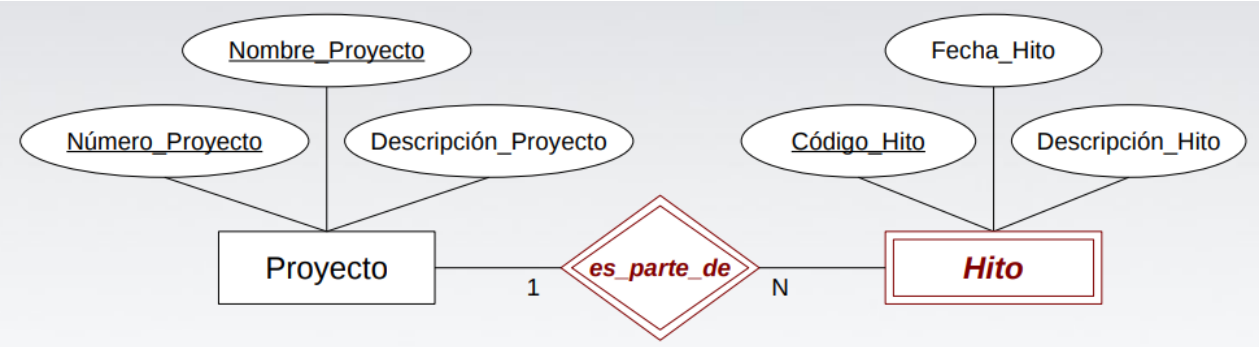
- Operación restringida (RESTRICT): el borrado o modificación de tuplas de la relación que contienen la clave primaria referenciada sólo se permite si no existen tuplas con dicha clave en la relación que contiene la clave ajena.
  - Por ejemplo, en el caso anterior esto implicaría que sólo podemos borrar una editorial en el que no haya ningún libro de esa editorial, si no, el sistema impediría el borrado.
- Operación con transmisión en cascada (CASCADE): el borrado o modificación de tuplas de la relación que contienen la clave primaria referenciada lleva consigo el borrado o modificación en cascada de las tuplas de la relación que contienen la clave ajena.
  - Por ejemplo, en el caso anterior al modificar el código de una editorial, se debe modificar también dicho código en todos los libros de nuestra Base de Datos que sean de dicha editorial.
- Operación con puesta a nulos (SET NULL): el borrado o modificación de tuplas de la relación que contienen la clave primaria referenciada lleva consigo “poner a nulos” los valores de las claves ajenas de la relación que referencia.
  - Por ejemplo, en el caso anterior cuando se borra una editorial, a todos aquellos libros que sean de esa editorial y que se encuentran en la relación LIBROS se les coloca el atributo cod_edit a nulo. Esta opción sólo es posible cuando el atributo que es clave ajena admite el valor nulo.
- Operación con puesta a valor por defecto (SET DEFAULT): el borrado o modificación de tuplas de la relación que contienen la clave primaria referenciada lleva consigo “poner al valor por defecto a la clave ajena de la relación que referencia; valor por defecto que habrá sido definido al crear la tabla correspondiente. Esto lo hace incompatible con una restricción de clave única (UNIQUE).
- Operación que desencadena un procedimiento de usuario: el borrado o modificación de tuplas de la relación que contienen la clave primaria referenciada lleva consigo la puesta en marcha de un procedimiento definido por el usuario.

En todo caso la operación que se desencadene será independiente para operaciones de actualización y de borrado. Por ello se podría elegir que ante un borrado se ponga a nulo y que bajo una actualización sea de transmisión en cascada.

2. Reglas de transformación

La transformación del modelo Entidad / Relación al modelo relacional se basa en las siguientes reglas:

- Toda entidad fuerte se transforma en una tabla (también llamada relación).
- Todo atributo se transforma en una columna dentro de la tabla.
- Los atributos identificadores se convierten en claves primarias (PK) y los identificadores alternativos (alt) se transforman en claves alternativas o de restricción única (U).
- En caso de las entidades débiles que dependen de otra entidad a través de una relación, la regla es que se propaga la clave de la entidad fuerte a la entidad débil, y esta a su vez formará parte de la clave.



El resultado relacional es el siguiente:

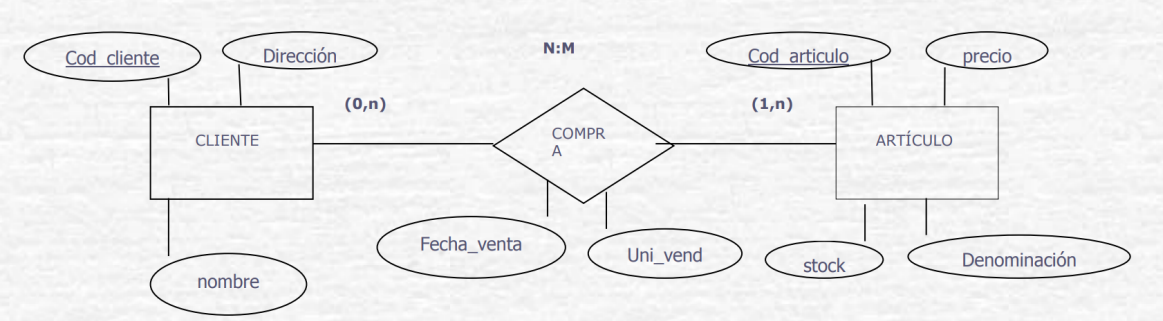
Proyecto (num_proy, nombre_proy, descripción, (PK: num_proy, nombre_proy))

Hito (num_proy (FK:Proyecto), nombre_proy (FK:Proyecto), cod_hito, fecha, descripción, (PK:num_proy, nombre_proy, cod_hito))

2.1. Reglas de transformación N:M

Toda relación N:M se transforma en una tabla que tendrá como clave primaria la concatenación de los atributos clave de las entidades que asocia.

Ejemplo Página 22



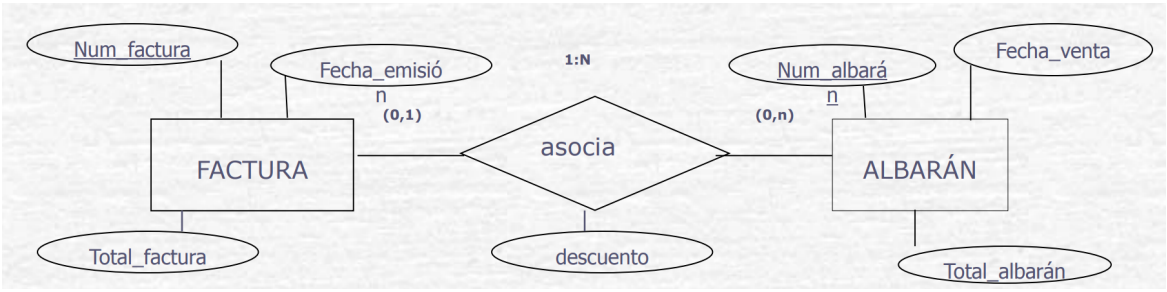
La transformación resulta en tres relaciones.

- Clientes: (cod_cliente(pk), nombre, dirección)
- Artículos: (cod_articulo (pk), precio, stock, denominación)
- Compras: (cod_cliente(fk:clientes), cod_articulo(fk:articulos), uni_vend, fecha_venta, (PK:cod_cliente,cod_articulo))

2.2. Reglas de transformación 1:N

En el caso de las cardinalidades 1:N existen dos soluciones para la transformación:

- Transformar la relación en una tabla. Si la relación es opcional, esto es relaciones con correspondencias (0,1) - (0,N) o (0,1) - (1,N), se trata como en la relación N:M creando una nueva tabla con los dos atributos identificadores de las entidades referenciadas como claves ajenas, pero cuya clave primaria es la clave de la entidad del lado N. También aplicaría esta norma si en el futuro la tabla resultante se puede transformar en una relación N:M.



El modelo relacional resultante es:

Facturas: (num_factura (PK), fecha_emisión, total_factura)

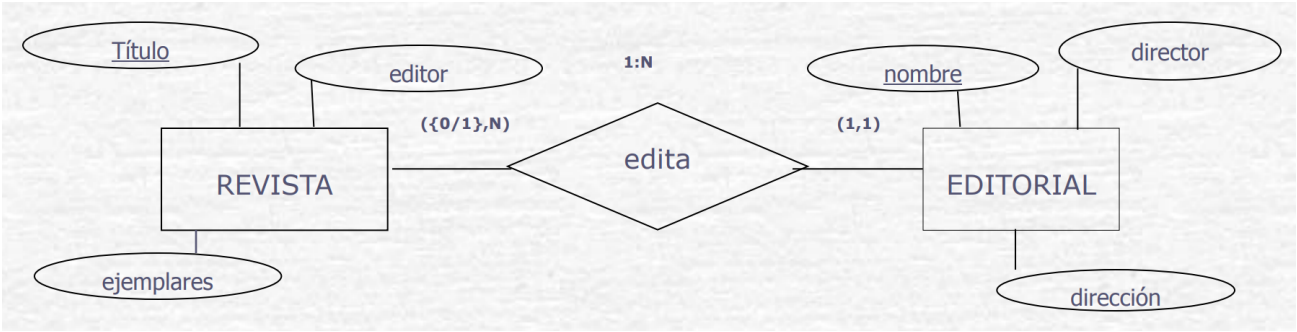
Albaranes: (num_albarán (PK), fecha_venta, total_albarán)

Fat-Alb: (num_factura (FK:Factura), num_albarán (FK:Albaranes)(PK), descuento)

Prestar atención a que num_albarán es clave principal, y tanto num_albarán como num_factura son claves ajenas de Albarán y Factura respectivamente.

- Propagar la clave: este caso se aplica cuando la relación es obligatoria, esto es con relaciones (1,1) - (0,N) o (1,1) - (1,N). Se propaga el atributo identificador de la entidad con correspondencia (1,1) a la nueva tabla correspondiente a la entidad del lado (0,N) o (1,N). La relación entre las entidades desaparece, y si esta tenía atributos propios se propagan también, pero como atributos normales de la entidad del lado (1,N).

Por ejemplo:



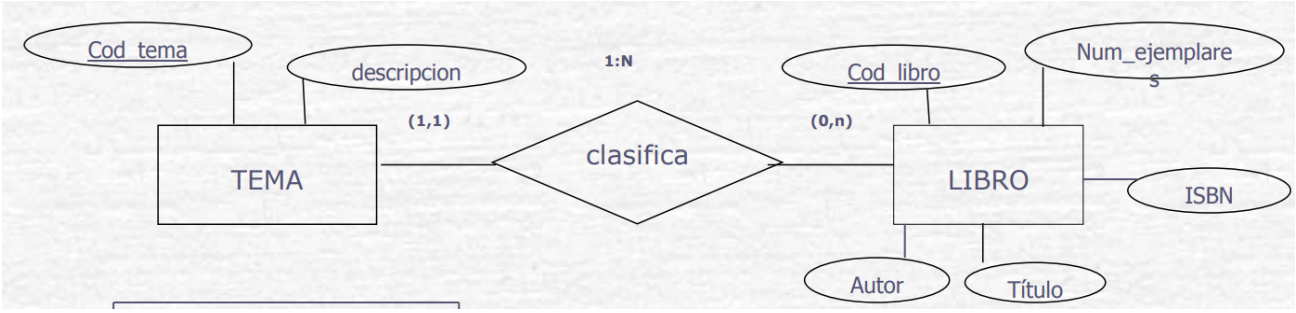
El modelo relacional resultante es:

Editoriales: (nombre (PK), director, dirección)

Revistas: (título (PK), editor, ejemplares, nombre(FK:Editorial))

Véase que nombre en Revistas es clave ajena que referencia a Editoriales.

Otro ejemplo es el siguiente:



El resultado relacional es el siguiente:

Temas: (cod_tema (PK), descripción)

Libros: (cod_libro (PK), autor, ISBN, título, num_ejemplares, cod_tema (FK: Temas))

En este caso se propaga la clave de la entidad Temas a la entidad Libros.

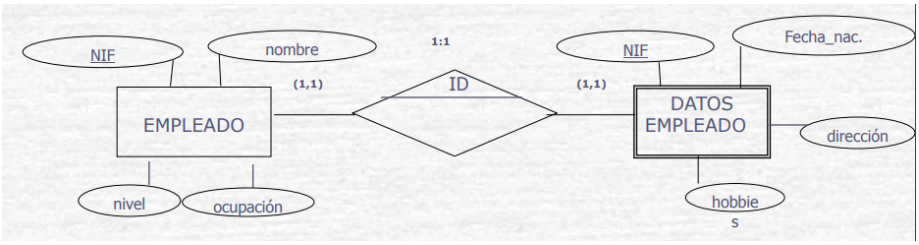
2.3. Reglas de transformación 1:1

Con cardinalidades 1:1 hay tres casos:

- Ambas entidades participan de forma obligatoria: (1,1) - (1,1)
- Una de las entidades participa de forma no obligatoria: (1,1) - (0,1)
- No hay obligatoriedad: (0,1) - (0,1)

Participación obligatoria

Cuando tenemos la participación obligatoria en ambas entidades (1,1) - (1,1) hay varias opciones según sean las entidades. Si las entidades tienen el mismo atributo como identificador, se creará una única relación con todos los demás atributos de las relaciones.



El resultado será:

Empleado: (NIF (PK), nombre, nivel, ocupación, fecha_nac, dirección, hobbie).

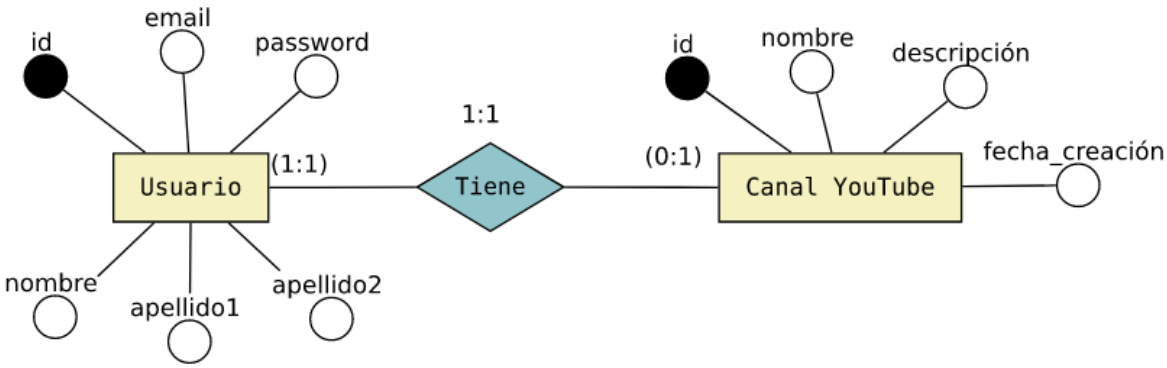
En el caso de que fuesen diferentes, mantendríamos las relaciones de ambas entidades, y una de ellas tendría una clave ajena de la otra.

Empleado: (NIF (PK), nombre, nivel, ocupación)

Datos_Empleado(NSS (PK), NIF(FK: Empleado), fecha_nac, dirección, hobbie)

Parcialmente obligatoria

Si la participación de una de las entidades no es obligatoria, (1,1) - (0,1) se procede de otra forma diferente. Se crearán dos relaciones, una por cada entidad. Posteriormente se propaga la clave de la entidad obligatoria (1,1) a la relación que es opcional (0,1):



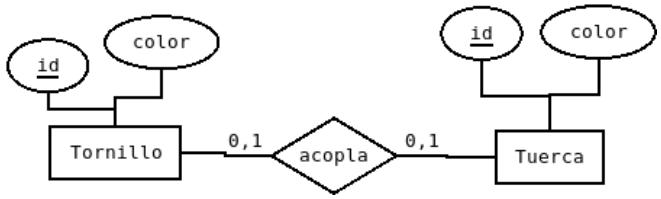
En este caso el esquema relacional es el siguiente:

Usuario: (id (PK), email, password, nombre, apellido1, apellido2)

Canal_Youtube: (id (PK), usuario (FK:Usuario)(U), nombre, descripción, fecha_creación)

Relación no obligatoria

En el caso de la correspondencia (0,1) - (0,1) se procederá a crear una nueva tabla para la relación, además de las dos entidades. Por otro lado, la clave primaria será una de las dos claves ajenas, mientras que la otra será clave candidata (lo que sería en SQL la restricción Unique).



De esta forma se transformaría en:

Tornillo: (id (PK), color)

Tuerca: (id (PK), color)

Acopla: (id_tornillo (PK) (FK: Tornillo), id_tuerca (U) (FK: Tuerca))

Notar que excepto en el caso de que en Entidad / Relación se propusiese un atributo como identificador alternativo, no hay ninguna norma de transformación que proponga claves candidatas más que en este caso. Como posteriormente en SQL se indicará con UNIQUE, se denota con (U) a la clave candidata.

2.4. Reglas de transformación reflexivas

Relaciones Reflexivas o recursivas son relaciones binarios en las que participa un solo tipo de entidad. En el proceso de convertir una relación reflexiva a tabla hay que tener en cuenta sobre todo la cardinalidad. Lo normal es que toda relación reflexiva se convierta en dos tablas, una para la entidad y otra para la relación. Se puede presentar los siguientes casos:

- Relación 1:1, Se actúa de forma similar a los casos de relaciones binarias 1:1.
  - En el caso de que el resultado sea propagar la clave, la clave de la entidad se repite, con lo que la tabla resultante tendrá dos veces ese atributo, una como clave primaria y otra como clave ajena de ella misma.
  - En el caso de crear una nueva tabla, se tendrá en esta dos claves ajenas a la misma tabla que resulte de la entidad. Para saber cuál es la clave se procede de la misma forma que en la binaria, normalmente una de las dos es clave y en el caso de (0,1) - (0,1) la otra será clave candidata (UNIQUE).
- Si la relación es 1:N, podemos tener dos casos:
  - Caso de que la entidad muchos sea siempre obligatoria se procede como en el caso 1:1.
  - Si no es obligatoria, se crea una nueva tabla cuya clave será la de la entidad del lado muchos, y además se propaga la clave a la nueva tabla como clave ajena.
- Si es N:N se trata igual que en las relaciones binarias. La tabla resultante de la relación contendrá dos veces la clave primaria de la entidad del lado muchos, más los atributos de las relaciones si los hubiera. La clave de esta nueva tabla será la combinación de las dos.

Ejemplo:

Consideramos la relación Empleado-dirige-empleado. Un empleado puede dirigir a muchos empleados o a ninguno si es el que dirige. Y un empleado es dirigido por un director o por ninguno si el es el que dirige. En este caso no ha obligatoriedad en la entidad muchos. El Modelo E/R es:

La tabla dirige tiene como clave primaria el código de empleado, que a su vez será clave ajena. Además se le añade el código de empleado pero, en este caso, tendrá el papel de director, que a su vez será clave ajena a la tabla EMPLEADO. El resultado será:

Empleado: (cod_emple (PK), dirección, teléfono, nombre)

Dirige (cod_emple (PK) (FK:Empleado), cod_emple-director(FK:Empleado))

Si representamos el estado completo de la relación, con datos, este sería un ejemplo:

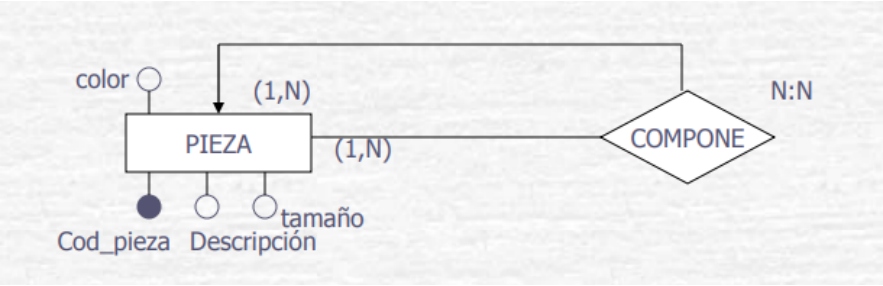
Empleados

Cod_emp
E1
E2
E3
E4
E5
E6
E7

Dirige

cod_emp	cod_emp_jefe
E2	E1
E3	E1
E4	E2
E5	E2
E6	E3
E7	E3

Se puede ver cómo el empleado 1 (E1) es jefe de E2 y E3. Estos a su vez son jefes de E4 a E7, dos cada uno. Al ser clave primaria el empleado "no jefe", este solo podrá tener un jefe.
Otro ejemplo es la relación una pieza se compone de muchas piezas, que a su vez están compuestas de otras piezas, es decir Pieza_Compone_Pieza. El Modelo E/R es:

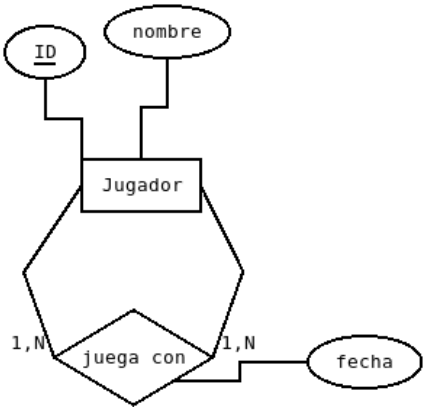


El modelo relacional resultante es:

Pieza: (cod_pieza (PK), descripción, tamaño, color)

ComponePieza: (cod_pieza_compone (FK:Pieza), cod_pieza_compuesta (FK:Pieza), (PK: cod_pieza_comp,cod_pieza))

Otro ejemplo es el siguiente:



El modelo relacional resultante es:

Jugador: (ID (PK), nombre)

Juega_con: (ID_1 (FK:Jugador), ID_2 (FK:Jugador), (PK: ID_1,ID_2),fecha)

Un ejemplo de tablas es el siguiente:

Jugador

ID
J1
J2
J3
J4
J5
J6
J7

Juega_con

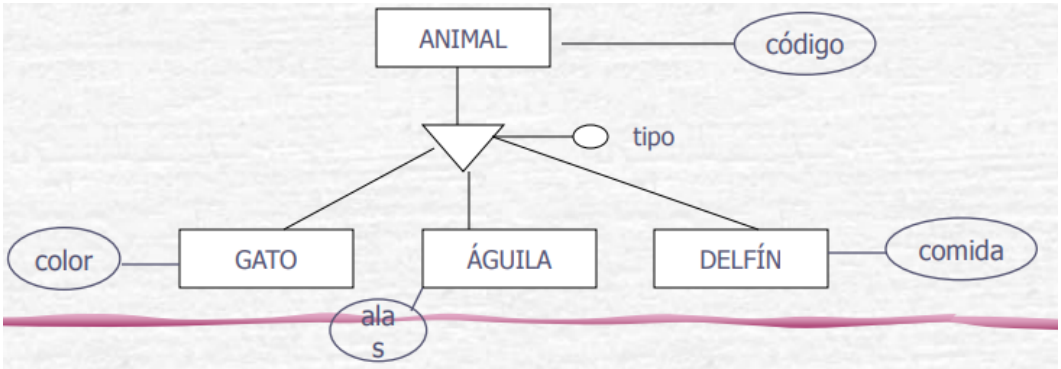
ID1	ID2	fecha
J1	J2	1-1-2000
J1	J3	2-1-2000
J2	J3	3-1-2000
J2	J4	4-1-2000
J3	J5	5-1-2000
J4	J5	6-1-2000
J4	J1	6-1-2000

Se puede observar cómo al ser ID1 e ID2 identificadores, pueden repetirse individualmente: J1 aparece dos veces en ID1 (incluso otra vez en ID2), y por otro lado J3 aparece dos veces. Lo que no podrá ser posible es que aparezca J1-J2 dos veces. Por otro lado, nada impide que aparezca J1-J2 y J2-J1. Esto ya depende de si tiene sentido: juega en casa o juega fuera si es como el fútbol. Pero si no tuviese sentido no podría ser restringido de ninguna forma más que con disparadores / triggers.

2.5. Reglas de transformación generalizaciones/especificaciones

El modelo relacional no dispone de mecanismos fáciles de usar que permitan la representación de relaciones jerárquicas, por lo tanto se tienen que eliminar. Para pasar estas relaciones al modelo relacional se aplicará una de las siguientes reglas:

- Integrar todas las entidades en una única eliminando a los subtipo. Esta nueva entidad contendrá todos los atributos del supertipo, todos los de los subtipos, y los atributos discriminativos para distinguir a qué subentidad pertenece cada atributo. Todas las relaciones se mantiene con la nueva entidad.
- Eliminación del supertipo, transfiriendo los atributos del supertipo a cada uno de los subtipos. El atributo de la relación se puede o no transferir. Las relaciones del supertipo se consideran para cada uno de los subtipos. Solo puede ser aplicada para jerarquías totales y exclusivas.
- Insertar una relación 1:1 entre el supertipo y cada uno de los subtipos. Los atributos se mantienen y cada subtipo se identificará con la clave ajena del supertipo. El supertipo mantendrá una relación 1:1 con cada subtipo. Los subtipos mantendrán, si la relación es exclusiva, la cardinalidad mínima 0, y si es inclusiva 0 ó 1.



El resultado a modelo relacional aplicando la primera norma es el siguiente:

Animales: (codigo (PK), tipo, color, alas, comida)

El resultado aplicando la segunda norma es el siguiente:

Gatos: (codigo (PK), color).

Águila: (código (PK), alas).

Delfines: (código (PK), comida).

El resultado aplicando la tercera norma es el siguiente:

Animales (código (PK))

Gatos: (codigo (FK:Animales) (PK), color).

Águila: (código (FK:Animales) (PK), alas).

Delfines: (código (FK:Animales) (PK), comida).

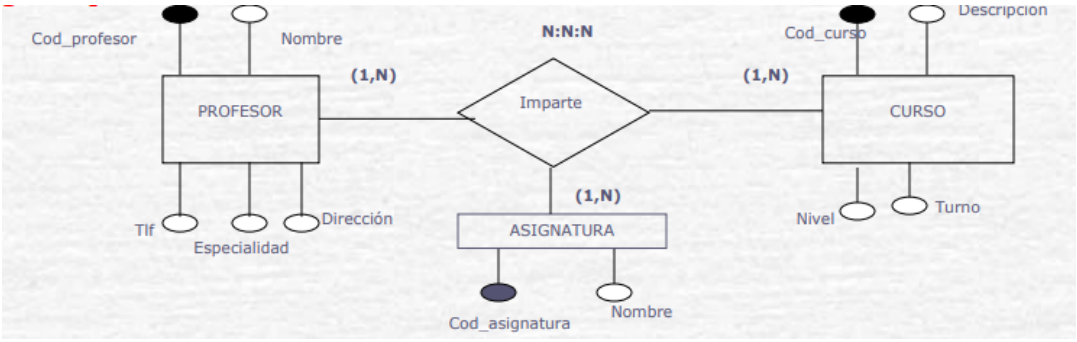
Cada forma tiene sus ventajas y desventajas y deberá aplicarse según la conveniencia del caso: importante es tener en cuenta si la generalización / especificación es total o exclusiva, si cada entidad tiene atributos propios, o más importante aún: si cada entidad tiene por su cuenta relaciones con otras entidades.

2.6. Reglas de transformación N-arias

En relaciones que apliquen sobre tres o más entidades el caso será siempre crear una nueva relación que tendrá como claves ajenas todas las claves primarias de las entidades que une. Si además tuviese atributos, se añadirían a la nueva relación.

En el caso de la clave primaria, estará formada por aquellas claves ajenas cuya correspondencia máxima sea N.

Se muestra en los dos siguientes ejemplos:



Y el diagrama relacional resultante es:

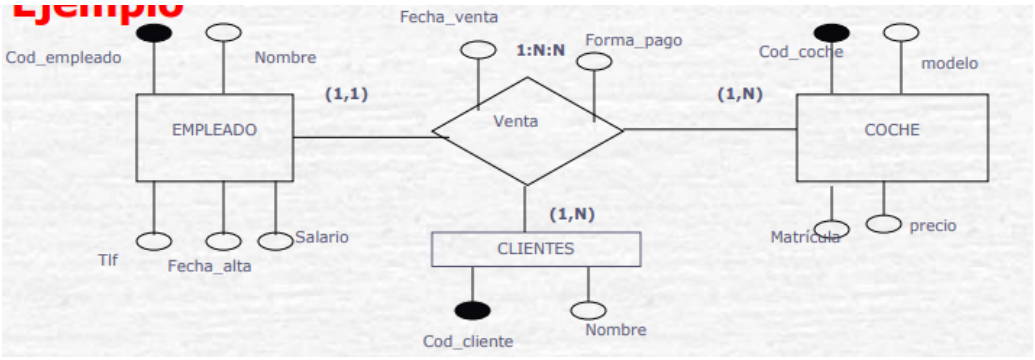
Profesor: (cod_prof (PK), nombre, tlf, especialidad, dirección)

Curso: (cod_curso (PK), descripción, nivel, turno)

Asignatura: (cod_asignatura (PK), nombre)

Imparte: (cod_prof (FK:Profesor), cod_curso (FK: Curso), cod_asig (FK: Asignatura), (PK:cod_prof, cod_curso, cod_asig))

El siguiente ejemplo tiene una de las correspondencias (1,1):



Y el resultado es el siguiente:

Empleado: (cod_emp (PK), nombre, tlf, dirección, salario)

Coche: (cod_coche (PK), modelo, matricula, precio)

Clientes: (cod_cliente (PK), nombre)

Venta: (cod_emp (FK: Empleado), cod_coche (FK: Coche), cod_cli (FK: Cliente), fecha_venta, forma_pago, (PK: cod_coche, cod_cli))