2
Diseño lógico: Modelo Relacional

2.1 INTRODUCCIÓN AL MODELO RELACIONAL

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

- **Sencillez y uniformidad**: Los usuarios ven la base de datos relacional como una colección de tablas, y al ser la tabla la estructura fundamental del modelo, éste goza de una gran uniformidad, lo que unido a unos lenguajes no navegacionales y muy orientados al usuario final, da como resultado la sencillez de los sistemas relacionales.
- **Sólida fundamentación teórica**: Al estar el modelo definido con rigor matemático, el diseño y la evaluación del mismo puede realizarse por métodos sistemáticos basados en abstracciones.
- **Independencia de la interfaz de usuario**: los lenguajes relacionales, al manipular conjuntos de registros, proporcionan una gran independencia respecto a la forma en la que los datos están almacenados.

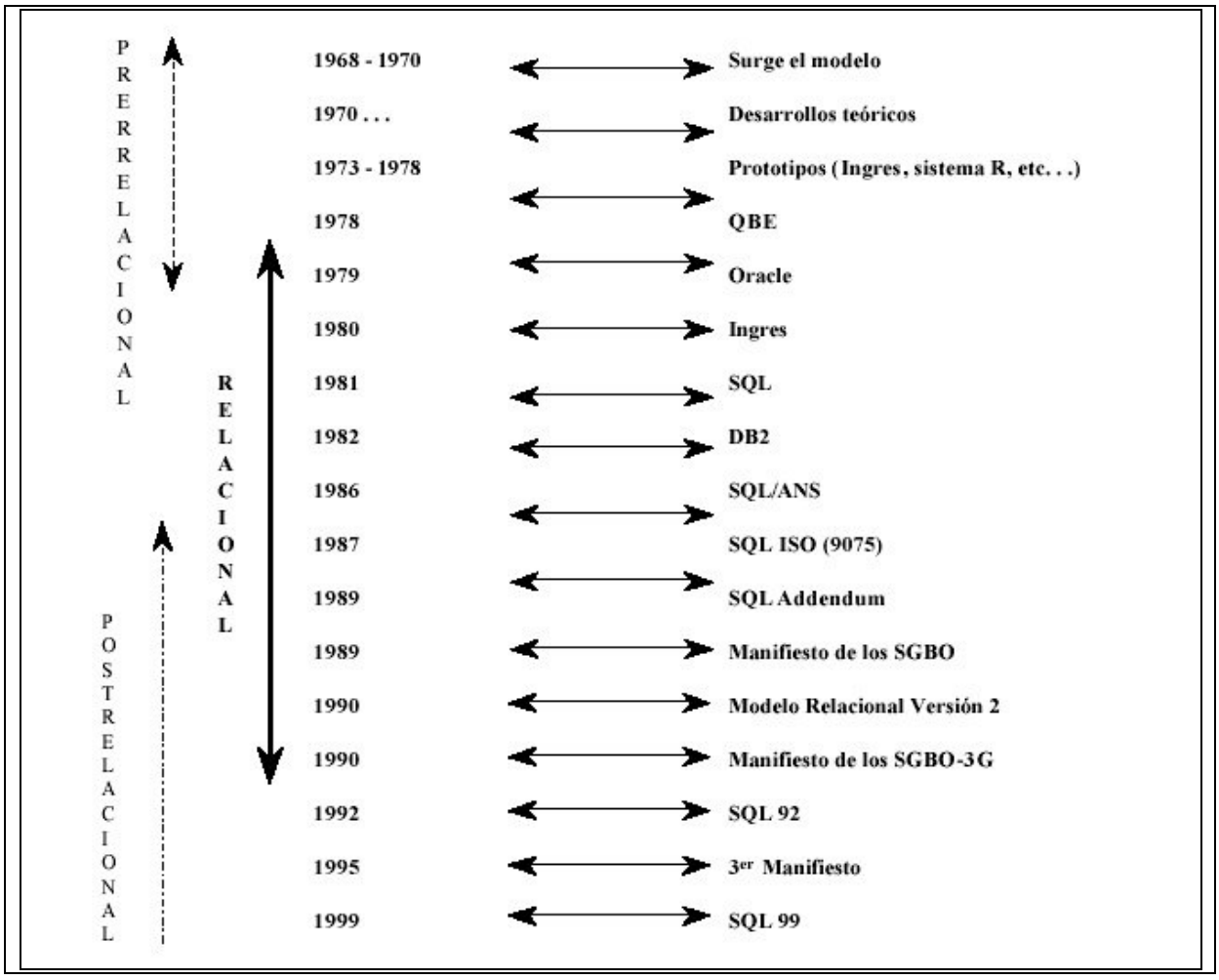


Figura 1.1 Evolución del modelo relacional

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

- **La prerrelacional -primera generación de BD's-**, en la cual los SGBD se soportan en los modelos Codasyl (en red) y Jerárquico.
- **La relacional -segunda generación de BD's-** donde los sistemas relacionales se van aproximando a su madurez y los productos basados en este modelo van desplazando poco a poco a los sistemas de primera generación, hasta conseguir una mayor cuota en el mercado de las bases de datos.
- **La postrelacional -tercera generación de Bd's-** en la que aparecen otros modelos de datos, en especial los orientados al objeto, que están en estos momentos intentando abrirse un hueco en el mercado de las bases de datos e integrándose como extensiones en los SGBD's previos de la generación relacional.

2.1.1 Elementos Básicos

Atributo

Los atributos de una relación se definen sobre dominios formados por un conjunto de valores homogéneos y atómicos. Representa las propiedades de la relación y se representa mediante una *columna*

Dominio

Es el conjunto válido de *valores* que toma un atributo. Es posible definir dominios por extensión, enumerando los posibles valores de los que consta el dominio, o binen indicando un rango de valores.

Tupla

Es una ocurrencia de la relación. Se representa mediante una *fila*

Relación

Una relación tiene un *nombre*, un *conjunto de atributos* que representan sus propiedades y un *conjunto de tuplas* que incluyen los valores que cada uno de los atributos toma para cada elemento de la relación.

La relación es el elemento fundamental del modelo relacional (de ahí el nombre del modelo), y se puede representar en forma de tabla de dos dimensiones, donde las columnas son los atributos de la relación y las filas son las tuplas. Así pues la relación *alumno* sería:

Atributo 1	Atributo 2	Atributo 3	....	Atributo n	
COD_MATRÍCULA	**NOMBRE**	**CIUDAD**		**COD_GRUPO**	
101	Juan Montero	Alcorcón	....	11	tupla 1
102	Alicia Cristóbal	Leganés	....	11	tupla 2
...	....	....	....	....	....
202	Ana Vallejo	Leganés	....	21	tupla n

Una relación no es una tabla. Existen diferencias entre ambas estructuras. La relación Alumno la representaremos de la siguiente forma:

ALUMNO(<u>cod_matricula</u>, nombre, ciudad, ..,*cod_grupo*)

2.1.2 Tipos de Claves

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

La propiedad de minimalidad implica que no se incluye ningún atributo innecesario. Una relación puede tener más de una clave candidata. En este caso se debe distinguir entre:

Clave Primaria (*Primary Key*):

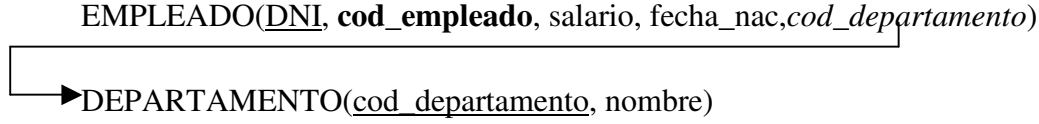
- Es la clave candidata que el usuario escoge para identificar las tuplas de la relación.
- Cuando sólo existe una clave candidata, ésta es la clave primaria (siempre existe clave primaria).

Claves Alternativas (*Alternative Key*):

- Las claves candidatas que no han sido escogidas como clave primaria.

Para relacionar unas relaciones con otras disponemos de la claves ajenas. Se denomina **clave ajena** de una relación R2 a un conjunto no vacío de atributos cuyos valores han de coincidir con los valores de la clave candidata de una relación R1. Las relacoines R1 y R2 pueden ser la misma relación. La clave ajena y la correspondiente clave candidata han de estar definidas sobre el mismo dominio.

El ejemplo siguiente la relación EMPLEADO tiene un clave primaria llamada *DNI*, una clave alternativa llamada *cod_empleado* y una clave ajena llamada *cod_departamento* que referencia a la clave primaria de la relación DEPARTAMENTO.



2.1.3 Restricciones del modelo relacional

Las **restricciones inherentes** vienen impuestas por el propio modelo de datos. En el caso del modelo relacional, una relación tiene unas propiedades intrínsecas que no tiene una tabla, y que se derivan de la misma definición matemática de relación, ya que, al ser un conjunto, en una relación:

- No puede haber dos tuplas iguales, es decir, es obligatorio una clave primaria que identifique cada una de las tuplas de forma única.
- El orden de las tuplas y el orden de los atributos no es relevante.
- Cada atributo sólo puede tomar un único valor del dominio sobre el cual está definido, es decir, no hay grupos repetitivos.
- Ningún atributo que forme parte de la clave primaria de una relación puede tomar un valor nulo, es decir desconocido o inexistente. Esto se conoce como la regla de integridad de entidad.

Los tipos de **restricciones semánticas** permitidos en el modelo relacional son:

- **Clave Primaria** (PRIMARY KEY): nos permite declarar un atributo o un conjunto de atributos como *clave primaria* de una relación. Sus valores no se podrán repetir ni se admitirán los nulos (o valores "ausentes").
- **Unicidad** (UNIQUE): los valores de un conjunto de atributos (uno o más) no pueden repetirse en una relación. Esta restricción permite la definición de claves alternativas.
- **Obligatoriedad** (NOT NULL): nos permite declarar si uno o varios atributos de una relación deben tomar un valor. El conjunto de atributos no admite valores nulos.
- **Integridad Referencial** (FOREING KEY): si una relación R2 (relación que referencia) tiene un descriptor (subconjunto de atributos) CA que referencia a una clave candidata CC de la relación R1 (relación referenciada), todo valor de dicho descriptor CA debe coincidir con un valor de CC o ser nulo.

  El descriptor CA es, por tanto, una clave ajena de la relación R2. Todo atributo de una clave primaria compuesta de una relación R2 que no está definido sobre un dominio compuesto, debe ser clave ajena de R2 referenciando a una relación R1, cuya clave primaria sea simple.

Editorial

Nombre_e	**Dirección**	**País**	**Ciudad**
Universal Books	Brown Sq.23	EEUU	Los Angeles
Rama	Canillas, 144	España	Madrid
McGraw-Hill	Basauri, 117	España	Madrid
Paraninfo	Virtudes, 7	España	Madrid

Libros

ISBN	**Título**	**...**	***Editorial***
0034523452	Int. Artificial		*Paraninfo*
1224344441	Concep. y Dis		*Rama*
2298108485	Turbo C++		*McGraw-Hill*
9876543255	Virus Informáticos		
0987654678	Sistemas Operativos		*Rama*

Figura 1.2 Ejemplo de integridad referencial

Además de definir las claves ajenas, hay que determinar las consecuencias que pueden tener ciertas operaciones (*borrado* y *modificación*) realizadas sobre tuplas de la relación referenciada; pudiéndose distinguir, según el estándar SQL92, las siguientes:

- **RESTRICT**: borrado o modificación restrictiva. Si existen tuplas de la relación hija relacionadas con la tupla de la relación padre sobre la que se realiza la operación, entonces no se permitirá llevar a cabo dicha operación.
- **CASCADE**: borrado o modificación en cascada. El borrado o modificación de una tupla en la relación padre ocasionará un borrado o modificación de todas las tuplas relacionadas en la relación hija.
- **SET NULL**: borrado o modificación con puesta a nulos. El borrado o modificación de una tupla de la relación padre pone a valor nulo la clave ajena de la tabla que referencia.

- **SET DEFAULT**: borrado o modificación con puesta a un valor por defecto. Pone un valor por defecto en la clave ajena de la tabla que referencia.

Ambos modos (de borrado y de modificación) son independientes, es decir, cada uno tomará una de las cuatro opciones por separado.

- **Restricciones de rechazo.** El usuario formula una condición mediante un predicado definido sobre un conjunto de atributos, tuplas o dominios, que debe ser verificado en toda operación de actualización para que el nuevo estado constituya una ocurrencia válida del esquema. En SQL92 existen dos clases:

  - **Verificación** (CHECK): Comprueba, en toda operación de actualización, si el predicado es cierto o falso y, en el segundo caso, rechaza la operación. La restricción de verificación se define sobre un único elemento (dentro de un CREATE TABLE) y puede o no tener nombre. Por ejemplo, en la tabla ALUMNOS la edad deber ser mayor de 18.

    CHECK N_HORAS > 30 en CURSO_DOCTORADO.

  - **Aserción** (ASSERTION): Actúa de forma idéntica a la anterior, pero se diferencia de ella en que puede afectar a varios elementos (por ejemplo, a dos tablas distintas). Por tanto, su definición no va unida a la de un determinado elemento del esquema y siempre ha de tener un nombre. Por ejemplo, solo podemos conceder una beca a los alumnos que la hayan solicitado anteriormente.

    ```
    CREATE ASSERTION CONCEDE_SOLICITA AS
    CHECK (SELECT Cod_Estudiante, Cod_Beca FROM CONCEDE) IN
    (SELECT Cod_Estudiante, Cod_Beca FROM SOLICITA));
    ```

- **Disparador** (*trigger*): Restricción en la que el usuario pueda especificar libremente la respuesta (acción) ante una determinada condición. Así como las anteriores reglas de integridad son *declarativas*, los disparadores son *procedimentales*, siendo preciso que el usuario escriba el procedimiento que ha de aplicarse en caso de que se cumpla la condición. Por ejemplo: si una beca es solicitada por más de 50 alumnos, se introduce un texto en una tabla de mensajes para que la persona que gestiona las becas considere si es necesario ofrecer más becas.

```
CREATE TRIGGER Comprobar_Matriculados
AFTER INSERT ON SOLICITA
DECLARE
        NUM_SOLICITUDES Number;
BEGIN
    SELECT COUNT(*) INTO NUM_SOLICITUDES FROM SOLICITA;
    IF NUM_SOLICITUDES > 50 THEN
        INSERT INTO MENSAJES VALUES ('Hay más de 50 solicitudes');
    END IF;
END Comprobar_Matriculados;
```

2.1.4 Notación

Un esquema relacional se representa mediante un grafo, conocido como grafo relacional. Básicamente se trata de un grafo dirigido cuyos nodos son las relaciones de la Base de Datos y los arcos representan las restricciones de clave ajena, y en el que aparecerán además de las distintas relaciones con sus atributos y las restricciones de clave ajena las restricciones de clave primaria, unicidad y obligatoriedad. Las convenciones utilizadas para la representación de este grafo son:

- El nombre de las tablas será representado en mayúsculas.
- Primero aparecerá el nombre de la relación y a continuación sus atributos entre paréntesis.
- Las claves primarias aparecen subrayadas.
- Las claves alternativas aparecen en negrita.
- Las claves ajenas están representadas en cursiva y referencian a la relación en la que son clave primaria mediante una flecha.
- Los atributos que pueden tomar valores nulos aparecen con un asterisco.
- Las opciones para la integridad referencias son:
  - B:C, Borrado en cascada.
  - B:N, Borrado con puesta a nulos.
  - B:D, Borrado con puesta a valor por defecto.
  - B:R, Borrado restringido.
  - M:C, Modificación en cascada.
  - M:N, Modificación con puesta a nulos.
  - M:D, Modificación con puesta a valor por defecto.
  - M:R, Modificación restringida.Reglas de transformación de un esquema E/R a un esquema relacional

2.2 LAS TRES REGLAS BÁSICAS EMPLEADAS EN LA TRANSFORMACIÓN DE UN ESQUEMA E/R EN UN ESQUEMA RELACIONAL SON:

- Toda entidad se transforma en una relación.
- Las interrelaciones M:N se transforman en una relación.
- Las interrelaciones 1:N dan lugar o bien a una propagación de la clave o bien a una relación.

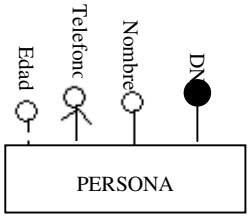
2.2.1 Transformación de entidades, atributos y dominios

Cada **Entidad** del esquema E/R dará lugar a una nueva relación cuya clave primaria es el identificador principal de la entidad. La relación se llamará igual que el tipo de entidad de donde proviene. Para su definición en SQL-2 se utiliza la sentencia CREATE TABLE.

Cada **Dominio** se representará mediante una sentencia CREATE DOMAIN en SQL-2. Un ejemplo de dominio para el atributo estado civil podría ser: CREATE DOMAIN e_civil AS CHAR(1) CHECK (VALUE IN ('S', 'C', 'V', 'D'))

Cada **Atributo** de una entidad se transforma en un atributo de la relación a la que ha dado lugar la entidad, aunque hay que tener en cuenta sus distintos tipos de restricciones.:

- *Atributos Univaluados*: Dan lugar a un atributo de la relación.
- *Atributos Multivaluados*: Dan lugar a una nueva relación cuya clave primaria es la concatenación de la clave primaria de la Entidad en la que se sitúa el atributo multivaluado más el nombre del atributo multivaluado. En ocasiones, si el atributo multivaluado no permite repeticiones es suficiente éste como clave primaria.



PERSONA(<u>DNI</u>, Nombre, Edad*)

B:C
M:C

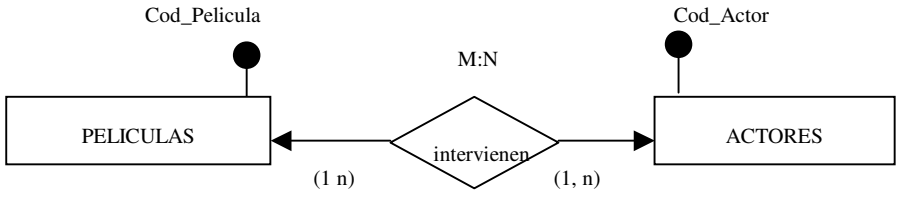
TELEFONOS(<u>*DNI*, Telefono</u>)

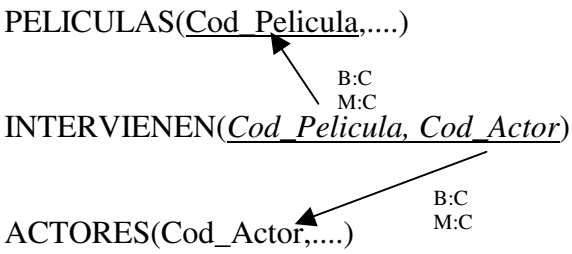
- *Atributos Obligatorios*: Atributos con restricción de NOT NULL.
- *Atributos Opcionales*: Atributos que pueden tomar valores nulos.
- *Atributos Identificadores:* Los atributos que son identificadores principales pasan a formar la clave primaria de la relación. En SQL se representan con la cláusula PRIMARY KEY dentro de la orden CREATE TABLE.
- *Atributos Identificadores Alternativos:* Se representan en SQL por medio de la claúsula UNIQUE dentro de la orden CREATE TABLE.
- *Atributos Derivados:* Atributos cuyo valor se obtiene como resultado de algún cálculo de otros atributos.
- *Atributos compuestos:* Se transforman en los atributos que los componen (no hay forma de representar una atributo compuesto en el modelo relacional).

2.2.2 Transformación de interrelaciones M:N

Un tipo de interrelación N:M se transforma en una relación que tendrá como clave primaria la concatenación de los identificadores principales de los tipos de entidad que asocia. Además, cada uno de los atributos que forman la clava primaria de esta tabla también son claves ajenas que referencian a las tablas en que se han convertido las entidades interrelacionadas (claves primarias).

Para cada clave ajena así obtenida deberá estudiarse cuales son los modos de borrado y modificación adecuados.

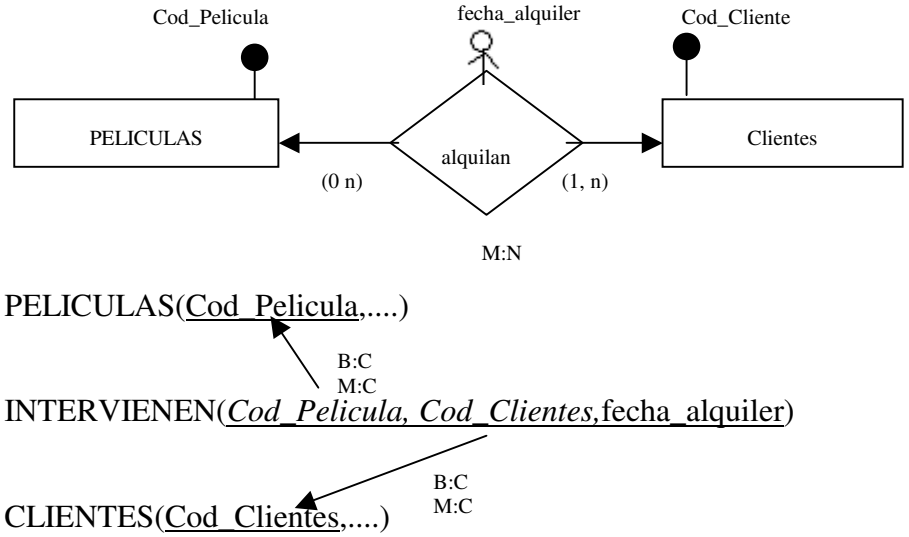




Las *cardinalidades mínimas* de las entidades participantes en la interrelación se pueden modelar utilizando restricciones, mediante una aserción.

Si la interrelación tiene atributos, éstos pasarán a formar parte de la relación. En caso de que la interrelación contenga un atributo multivaluado que no denote dimensión temporal, la clave de la relación deberá incluir también este atributo.

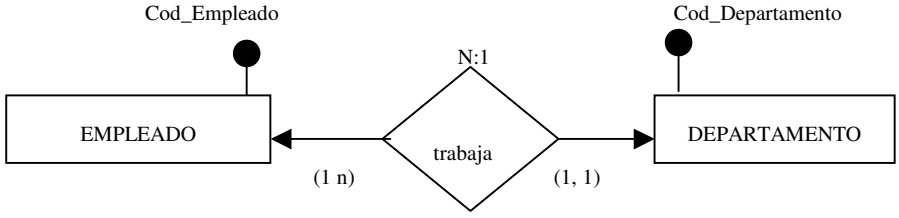
Sin embargo, en el caso de atributos con dimensión temporal (generalmente atributos que denotan fechas, horas o intervalos de tiempo), tanto si son multivaluados como univaluados, es necesario estudiar la semántica del universo de discurso con el fin de determinar cuáles serán los atributos que formen parte de la clave primaria de la relación a al que da lugar la interrelación.



2.2.3 Transformación de interrelaciones 1:N

En el caso de las interrelaciones 1:N existen dos posibles transformaciones:

- Propagar los identificadores principales del tipo de entidad que tiene de cardinalidad máxima 1 a la que tiene N (propagación de clave). Esta es la regla habitual.



EMPLEADO(Cod_Empleado, **...**,*cod_departamento*)

B:C
M:C

DEPARTAMENTO(cod_departamento, ...)

- Transformar la interrelación en una tabla como si se tratara de una interrelación N:M; pero ahora la clave primaria de la relación creada es sólo la clave primaria de la tabla a la que le corresponde la cardinalidad N. Esta opción se utiliza cuando:
  - El número de ejemplares interrelacionados de la entidad que propaga su clave es muy pequeño y, por tanto, existirían muchos valores nulos en la clave propagada.
  - Se prevé que la interrelación en un futuro se convertirá en una de tipo N:M.
  - La interrelación tiene atributos propios y no es deseable propagarlos (a fin de conservar la semántica).

Al hacer propagación de clave, la *cardinalidad mínima* de la interrelación con máxima 1 se puede modelar usando NOT NULL para el valor 1. Para la interrelación de cardinalidad máxima N es necesaria una restricción (check, assertion o trigger).

2.2.4 Transformación de interrelaciones 1:1

Una interrelación de tipo 1:1 es un caso particular de una 1:N, por lo que se pueden aplicar las dos opciones ya comentadas: crear una nueva relación o aplicar la propagación de clave, teniendo en cuenta que ahora la propagación de la clave puede efectuarse en ambos sentidos.

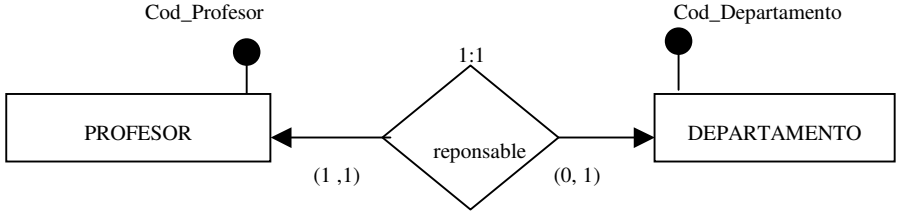
Los criterios para aplicar una u otra regla y para propagar la clave se basan en:

- las cardinalidades mínimas,
- recoger la mayor cantidad de semántica posible,
- evitar los valores nulos, o
- aumentar la eficiencia.

Si las entidades que se asocian poseen cardinalidades (0,1), suele ser conveniente transformar la interrelación 1:1 en una relación.

Si las entidades que participan en la interrelación poseen cardinalidades (0,1) y (1,1), conviene propagar la clave de la entidad con cardinalidades (1,1) a la relación resultante de la entidad con cardinalidades (0,1).

En el caso de que ambas entidades presenten cardinalidades (1,1), se puede propagar la clave de cualquiera de ellas a la relación resultante de la otra, teniendo en cuenta en este caso los accesos más frecuentes y prioritarios a los datos de las relaciones.



PROFESOR(Cod_Profesor, **.....**)

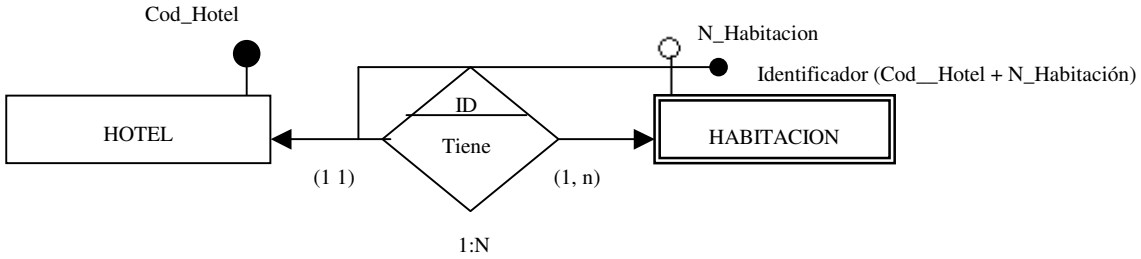
B:C
M:C

DEPARTAMENTO(Cod_Departamento, ...,*Cod_Profesor*)

2.2.5 Transformación de Dependencias en existencia y en identificación.

La manera de transformar una interrelación de este tipo es utilizar el mecanismo de propagación de clave, creando una clave ajena, con nulos no permitidos, en la tabla de la entidad dependiente, con la característica de obligar a una modificación y un borrado en cascada.

Además, en el caso de dependencia en identificación la clave primaria de la relación en la que se ha transformado la entidad débil debe estar formada por la concatenación de las claves de las dos entidades participantes en la interrelación.



HOTEL(Cod_Hotel, **.....**)

B:C
M:C

HABITACION(NumHabitación,*Cod_Hotel*,....1)

2.2.6 Transformación de Generalizaciones

Existen tres posibilidades de transformación de las jerarquías al modelo relacional:

- Englobar todos los atributos de la entidad y sus subtipos en una sola relación. En general, se debe adoptar esta solución cuando los subtipos se diferencian en muy pocos atributos y las interrelaciones que los asocian con el resto de las entidades del esquema sean las mismas para todos (o casi todos) los subtipos.
- Crear una relación para el supertipo y tantas relaciones como subtipos haya, con sus atributos correspondientes. Esta es la solución adecuada cuando existen muchos atributos distintos entre los subtipos y se quieren mantener de todas maneras los atributos comunes a todos ellos en una tabla.
- Considerar relaciones distintas para cada subtipo, que contengan, además de los atributos propios, los atributos comunes. Se elegirá esta opción cuando se dan las mismas condiciones que en el caso anterior –muchos atributos distintos- y los accesos realizados sobre los datos de los distintos subtipos siempre afectan a atributos comunes.

Aunque es posible elegir cualquiera de las tres estrategias para la transformación de un tipo y sus subtipos al modelo relacional, desde un punto de vista exclusivamente semántico la

opción segunda es la mejor; y desde el punto de vista de la eficiencia deberá tenerse en cuenta que:

En la primera opción el acceso a una fila que refleje toda la información de una determinada entidad es mucho más rápido (no hace falta combinar varias tablas).

El segundo opción es la menos eficiente aunque es la mejor desde un punto de vista exclusivamente semántico.

Con la opción tercera se aumenta la eficiencia ante determinadas consultas (las que afecten a todos los atributos, tanto comunes como propios, de un subtipo) pero se disminuye ante otras. Esta solución es en la que se pierde más semántica; además si existe solapamiento se introduce redundancia que debe ser controlada si se quieren evitar inconsistencias.

Se deberá elegir una estrategia u otra dependiendo de que sea la semántica o la eficiencia la que prime para el usuario en un momento determinado.

En los ejercicios propuestos se adoptará preferentemente la segunda opción. Es decir, se creará una relación por cada entidad participante en la jerarquía (una relación para el supertipo y otra para cada uno de los subtipos), de tal forma que el supertipo propaga su Identificador Principal a cada uno de los subtipos que pasan a identificarse por el mismo identificador (como clave ajena). La relación creada para el supertipo tendrá el *atributo discriminante* de la jerarquía.

También habrá que especificar las restricciones semánticas de totalidad/parcialidad y exclusividad/solapamiento.

La *totalidad* se controla prohibiendo las inserciones en el supertipo y con un disparador que ante una inserción en el subtipo inserte también en el supertipo (el atributo discriminante no podrá tomar valores nulos). En el caso de la *parcialidad* no es necesario ningún disparador (el atributo discriminante pude tomar valores nulos).

En cuanto a la *exclusividad* se requiere una aserción que compruebe que, si un ejemplar pertenece a uno de los subtipos, entonces no puede pertenecer a los demás. Si se permite *solapamiento* también es necesaria una aserción similar a la de la exclusividad pero que dé cabida a nuevos valores del atributo discriminante para los casos de solapamiento comprobando que las ocurrencias están en los subtipos adecuados.

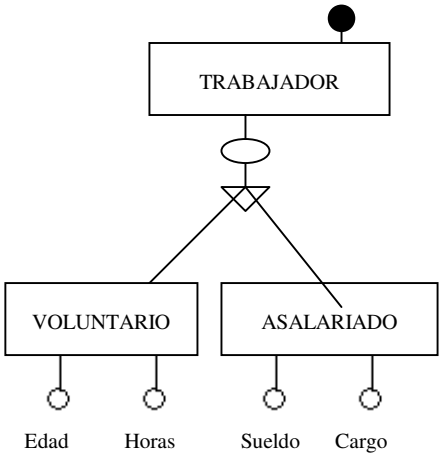
Por ejemplo, la exclusividad entre ESPECIALIZADO y APRENDIZ (subtipos de OPERARIOS) puede controlarse con una aserción de la siguiente forma.

```
CREATE ASSERTION controlar_exclusividad
Check ((Tipo='especializado' AND EXIST ESPECIALIZADO
AND NOT EXIST APRENDIZ
OR
(Tipo='aprendiz' AND EXIST APRENDIZ
AND NOT EXIST EXPECIALIZADO))
```

El *atributo discriminante* de la generalización podrá admitir valores nulos en el caso de que haya recubrimiento parcial y deberá declararse como NOT NULL si el recubrimiento es total.

El *atributo discriminante* constituirá un grupo repetitivo (multivaluado), si los subtipos solapan (generalización con solapamiento), debiendo, en este caso, separar este atributo en

una relación aparte que tendrá como clave la concatenación de la clave del supertipo con el atributo discriminante.



TRABAJADOR(Cod_Trabajador,....,Tipo)

B:C
M:C

VOLUNTARIO(*Cod_Trabajador*, Edad, Horas)

B:C
M:C

ASALARIADO(*Cod_Trabajador*, Sueldo, Cargo)

2.2.7 Transformación de interrelaciones de grado superior a dos

Las interrelaciones ternarias, cuaternarias, etc, se representan igual que las interrelaciones N:M, es decir, creando una nueva relación cuya clave primaria será la concatenación de los identificadores principales de los tipos de entidades participantes.

Para controlar las *cardinalidades mínimas y máximas* de cada entidad participante deberá recurrirse a restricciones.

2.2.8 Transformación de interrelaciones exclusivas.

En el caso de interrelaciones exclusivas, éstas se transforman como se ha visto en este mismo apartado, pero es necesario añadir un check que compruebe que si un ejemplar de la entidad participa ya en una ocurrencia de una interrelación, entonces no puede participar en ninguna ocurrencia de la otra interrelación.

2.3 PERDIDA DE SEMÁNTICA EN LA TRANSFORMACIÓN AL MODELO RELACIONAL

Existen algunas restricciones que es necesario controlar con mecanismos externos al modelo relacional. Aquí es necesario establecer la diferencia entre el modelo relacional y los Sistemas Gestores de Bases de Datos comerciales que no implementan el modelo relacional completo. Por ello, aunque en este documento se hable de checks y aserciones como mecanismos del modelo relacional para implementar cierta restricciones, estos mecanismos no siempre están disponibles en los sistemas gestores de BD, lo que implica que hay que recurrir a otros medios para recoger esas restricciones (por ejemplo, disparadores presentes en la BD,

procedimientos almacenados, aplicaciones externas, etc). Se dará aquí enumeración de las restricciones de los esquemas E/R que es necesario contemplar en la transformación al modelo relacional mediante checks, aserciones o disparadores:

- Cardinalidades mínimas de 1 en interrelaciones N:M y 1:N (excluyendo aquellas que se controlan con la restricción NOT NULL cuando se realiza propagación de la clave).
- Cardinalidades máximas conocidas en interrelaciones binarias N:M y 1:N e interrelaciones ternarias.
- Exclusividad en generalizaciones
- Inserción y borrado en las generalizaciones
- Atributos derivados.
- Exclusividad entre interrelaciones
- Atributos multivaluados obligatorios.