Unidad 2

"Modelo Relacional de una Base de Datos"

Gestión de Bases de Datos

Bases de Datos

IES Teis

Modelo lógico de datos

1 Introducción

Ya hemos visto que técnica (Diagramas Entidad/Relación) hay que utilizar para realizar el Modelo conceptual de datos. Ese Modelo conceptual se realiza en la fase de **Análisis** (dónde la pregunta sería "Qué" tenemos que hacer). En la siguiente fase, el **Diseño**, tendremos que realizar el modelo lógico de datos. En esta fase nos centraremos en "Cómo" tenemos que hacerlo.

MODELOS / FASES	Modelo de datos	Modelo de procesos	Modelo de eventos
Análisis	Modelo conceptual de datos	Análisis de procesos	Análisis de eventos
Diseño	Modelo lógico de datos	Diseño de procesos	Diseño de interfaces de usuario
Implementación	Modelo físico de datos	Codificación de procesos	Codificación de las interfaces de usuario

2 Objetivos y principales modelos lógicos de datos

Base de Datos (BBDD): Colección de datos interrelacionados almacenados sin redundancias perjudiciales o innecesarias.

Sistema de Gestión de Bases de Datos (SGBD): Software para describir, recuperar y manipular los datos almacenados en una BBDD.

El ***modelo lógico de datos*** está orientados a describir los datos teniendo en cuenta "*Cómo*" se tienen que implementar los datos en un SGBD. Por tanto el modelo lógico *depende del tipo de SGBD elegido*.

Los SGBD se apoyan en modelos teóricos. Los principales modelos teóricos en los que se basan los SGBD son:

- Modelo Jerárquico (En desuso)
- Modelo En red (En desuso)
- **Modelo Relacional**
- Modelo Orientado a Objetos

El modelo en que siguen la gran mayoría de los SGBD comerciales es el Modelo Relacional, por tanto, este será el modelo que vamos a estudiar.

Los SGBD que se apoyan en el modelo relacional se denominan Sistemas de Gestión de Bases de Datos Relaciónales (SGDBR) y las Bases de Datos que se crean con estos sistemas se denominarán Bases de Datos Relaciónales.

Algunos de los SGBDR que existen en el mercado son:

- Access
- SQL Server
- Oracle
- MySQL

En resumidas cuentas, como el modelo lógico de datos depende del SGBD elegido, y los SGBDR son los más utilizados, nosotros estudiaremos el Modelo Lógico Relacional de Datos.

3 Modelo LÓGICO RELACIONAL de datos

El modelo de datos consta de tres partes, como se ha visto en el Apartado 1 del Tema 4. En la fase de análisis de datos se ha obtenido el modelo Entidad-Relación. Modelo que corresponde al ámbito conceptual, es decir, al mundo de las ideas, el mundo real. En esta fase se indicó QUÉ datos hay que representar.

Una vez obtenido el modelo conceptual de datos, se pasa a la fase de CÓMO se van a representar esos datos, estamos por lo tanto ante el llamado **modelo lógico de datos**. En el cual se traducirá la estructura obtenida en la fase de análisis a un lenguaje más similar al que utiliza una base de datos fisicamente. En esta fase se obtendrá el **modelo relacional de datos**.

El modelo relacional es una técnica de diseño lógico de datos que representa CÓMO se deben de almacenar los datos en un sistema. Este modelo parte del modelo entidad-relación, correspondiente a la fase de diseño conceptual del sistema.

Este modelo supone una serie de ventajas que se expresan a continuación:

→ **Independencia física**: el modo en que se almacenan los datos no influye en la definición de los mismos, que se produce en esta fase.

→ **Independencia lógica**: Las operaciones sobre los datos no afectan a cómo los datos son tratados.

→ **Flexibilidad**: es flexible porque puede ofrecer diversos esquemas externos.

→ **Uniformidad**: en cuanto a la representación de los datos es una técnica uniforme, facilitando de este modo el empleo de los mismos

→ **Sencillez**: Es fácil de comprender y utilizar.

3.1 Conceptos del modelo relacional

En el modelo relacional la estructura básica es la **relación o tabla**, formada por un conjunto de **campos o atributos** (columnas) y de **tuplas o registros** (filas). El dominio será el conjunto de valores que puede tomar un atributo. El **grado** el número de atributos de una tabla y la **cardinalidad**, el número de tuplas de la misma.

NOTA: No confundir la cardinalidad y el grado del modelo relacional con la cardinalidad y el grado en los diagramas E/R. Simplemente tienen el mismo nombre pero en el Diagrama E/R significan cosas distintas.

Ejemplo, en el caso de la pizzería, las tablas del modelo relacional se corresponden con las entidades del modelo entidad-relación, en este caso una tabla podría ser repartidor, (ver ejemplo en Figura 17, Tema 4).

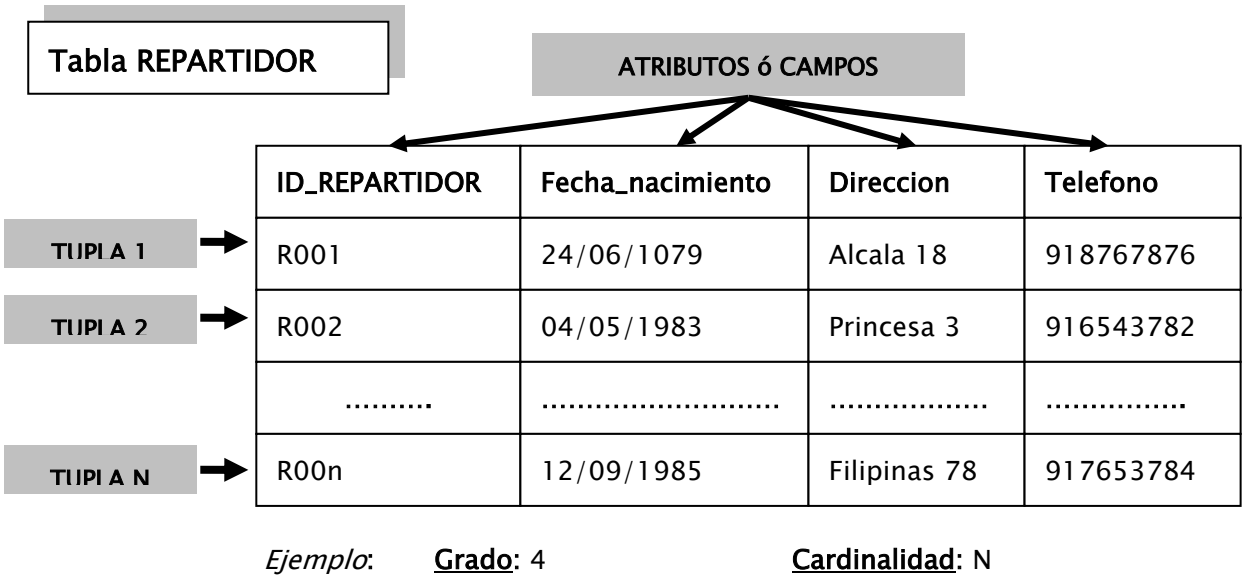


Figura 1. Tabla en el modelo relacional

3.2 Claves del modelo relacional

→ **Clave candidata** *(Candidate Key)*: Una clave candidata de una relación es uno o varios atributos que identifican unívoca y mínimamente cada tupla.

- 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 tiene que distinguir entre:

- **Clave primaria** *(Primary Key)*: aquella clave candidata que el usuario escoge (por consideraciones ajenas al modelo relacional) para identificar las tuplas de la relación.
- **Claves alternativas** *(Alternative Key)*: las claves candidatas que no han sido escogidas como primarias

→ **Clave ficticia**: es una clave inventada, normalmente suele ser un número consecutivo.

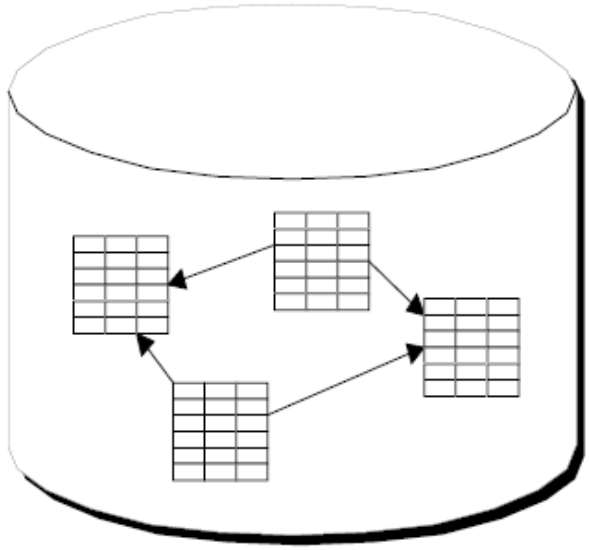
La correspondencia con el modelo entidad relación es la siguiente:

Modelo Entidad Relación	Modelo Relacional
Ocurrencia	Fila ó Tupla

Atributo	Columna
Entidad	Tabla ó Relación
Grado	Número de columnas
Cardinalidad	Número de Filas

3.3 Interrelaciones entre tablas

Una Base de Datos (relacional o de otro tipo) es una colección de datos (en nuestro caso relaciones o tablas) interrelacionados. Por tanto necesitamos asociar una relación (tabla) con otras. Para ello se utiliza lo que se denomina clave ajena.



Clave ajena *(Foreign Key)*: Se denomina clave ajena de una relación R2 a uno o varios atributos cuyos valores han de coincidir con los valores de la clave primaria de una relación R1.

- ✓ R1 y R2 pueden ser la misma relación (o Tabla).
- ✓ La clave ajena y la primaria deben estar definidas sobre los mismos dominios.

Representa las interrelaciones entre tablas

A continuación veremos un ejemplo de 2 relaciones (Tablas) con claves primarias y claves ajenas:
(En este ejemplo se representarán las relaciones por EXTENSIÓN y por INTENSIÓN).

Clave primaria

NUMDEPT	NOMDEPT	PRESUPUESTO
D1	Marketing	1000
D2	Desarrollo	1200
D3	Investigación	500

DEPARTAMENTO

Clave primaria Clave ajena Clave alternativa

NUMEMP	APELLIDO	NUMDEP	SALARIO	NIF
E1	López	D1	40	52.123.456-X
E2	Fernández	D2	42	40.123.456-Y
E3	Martínez	D3	50	30.123.456-Z
E4	Sánchez	D2	34	20.123456-A

EMPLEADO

Figura 2. Representación de la clave ajena por EXTENSIÓN de relaciones

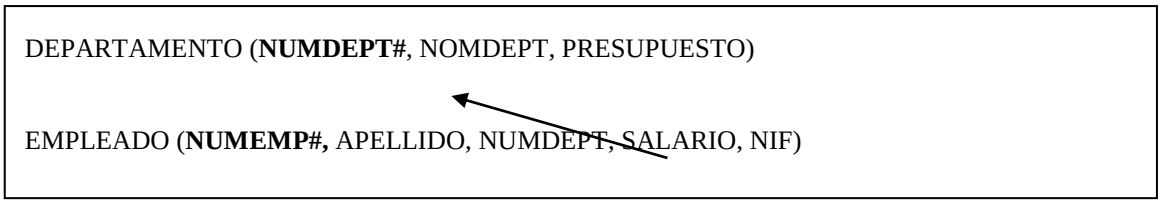


Figura 3. Representación de la clave ajena por INTENSIÓN de las relaciones

Según este ejemplo. ¿Cuál sería el nombre del departamento al que pertenece el empleado Martínez?

3.4 Restricciones del modelo relacional

Se pueden distinguir entre restricciones inherentes al modelo relacional y restricciones de usuario, las cuales se muestran más detalladamente en los siguientes apartados.

3.4.1 Restricciones Inherentes

Son las que están impuestas por el propio modelo relacional y son las siguientes:

→ Tener definida una clave primaria, esto se debe a que no se admiten duplicados, no podrán existir dos tuplas idénticas en una tabla, con la existencia de una clave primaria se garantiza esto. Restricción de clave primaria se impone una vez en cada tabla. Una columna o conjunto de columnas se definen como primary key.

→ El orden de los campos (atributos) no es significativo. Es decir, da igual el orden en que estén almacenados cada uno de ellos.

→ El orden de las tuplas (filas) no es significativo.

→ No se admiten atributos multivaluados (es decir, cada atributo tendrá un solo valor).

3.4.2 Restricciones de Usuario

Son aquellas que nos permiten recoger con la mayor fidelidad posible el mundo real objeto de nuestro diseño.

→ **Clave primaria (PRIMARY KEY)**: permite definir ***un*** atributo o ***un conjunto*** de atributos como **clave primaria** de una relación.

→ **Unicidad (UNIQUE):** obliga a que los valores de un/ un conjunto de atributos (uno o más) no pueden repetirse en una relación. Esta restricción permite la definición de **claves alternativas**. El diseñador de la base de datos cuando quiere definir una clave alternativa le impone a la columna las dos restricciones: obligatoriedad y unicidad.

→ **Obligatoriedad ó de integridad de entidad (NOT NULL):** no se admiten valores nulos, es decir, el valor de un atributo será siempre un valor que esté dentro del dominio al que pertenece el atributo.

→ **Integridad referencial (FOREING KEY):** El valor de una clave ajena tiene que ser un valor que exista en la tabla que referencia o ser nulo (si se admiten nulos). La integridad referencial se puede mantener de distintos modos, **Operación Restringida**, **Operación en Cascada**, **Puesta a Nulos** y **Puesta a valor por defecto**. Estos se explican a continuación con más nivel de detalle.

La integridad referencial trata la forma en la que los datos de dos o más tablas se deben relacionar para no atentar contra la integridad de la base de datos.

Para explicar la integridad referencial usaremos el siguiente ejemplo:

EJEMPLO:

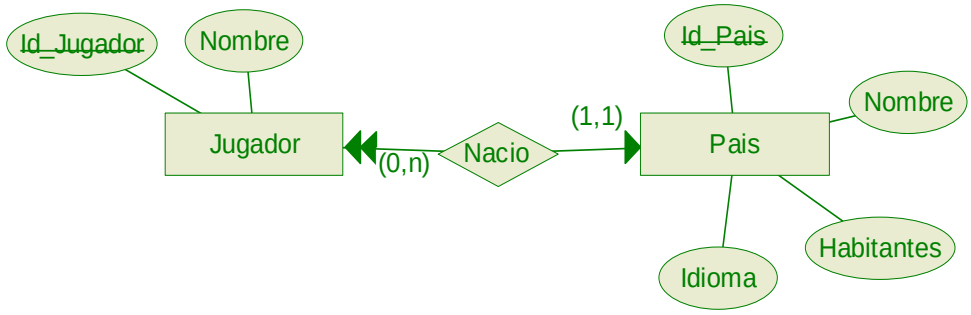


Figura 4. Ejemplo de Entidad Relación (JUGADORES DE FUTBOL)

Las tablas (relaciones) que se corresponden con este diagrama entidad relación en el modelo relacional serán las siguientes:

Tabla (Relación): JUGADORES		
Id_Jugador	**Nombre**	**Id_Pais**
J001	Raul Lopez	ESP
J002	Robinho	BRA
J003	Zinedine Zidane	FRA
J004	Beckam	ING

** En esta tabla, **Id_Pais**, actúa como clave ajena. Relacionando a la tabla con JUGADOR.*

Tabla (Relación): PAISES			
Id_Pais	**Nombre**	**Habitantes**	**Idioma**
ESP	ESPAÑA	9.000.000	ESPAÑOL
ING	INGLATERRA	12.000.000	INGLES
FRA	FRANCIA	11.345.655	FRANCÉS
BRA	BRASIL	23.098.476	PORTUGUÉS
HOL	HOLANDA	5.000.765	HOLANDÉS

Ejemplo: En el ejemplo de un jugador, si interpretamos la relación de que un jugador pueda poseer varias nacionalidades tendríamos:

JUGADOR (id_jugador#, nombre)

PAIS (id_pais#,nombre, habitantes,idioma)

POSEE_NACIONALIDAD (id_pais#,id_jugador#)

Si tenemos una tupla de la tabla POSEE_NACIONALIDAD con los siguientes valores:

(ESP, JUG_1)

El jugador "JUG_1" tiene que existir en la tabla JUGADOR previamente y el pais "ESP" tiene que existir previamente en la tabla PAIS. Esto se debe a que en la tabla POSEE_NACIONALIDAD, el id_jugador# y el id_pais# actúan como claves ajenas con respecto a las claves primarias de las tablas JUGADOR y PAIS, respectivamente.

La existencia de integridad referencial tiene consecuencias sobre las operaciones de modificación de la clave y borrado de las tuplas, por lo que se debe de escoger entre una de las siguientes acciones que controlan la consistencia de los datos cuando se produce una de estas operaciones de borrado o modificación:

(Para los ejemplos de estar restricciones vamos a utilizar el diagrama entidad-relación de la Figura 4., al que le corresponden los datos de las tablas que acompañan a la figura.). Ver notación respecto a Borrado y Modificación en el cuadro siguiente:

B/M: Borrado modificación.

R1: Relación o tabla que referencia (es la que contiene una columna definida con FK)

R2: Relación o tabla referenciada

BR, MR: Borrado Restringido, Modificación Restringida

BC, MC: Borrado en Cascada, Modificación en Cascada (Cascade)

BN, MN: borrado a Nulo (Set Null)

BD, MD: Borrado por defecto (Set Default)

- **operación restringida (NO ACTION):** solo se permite borrar (BR) o modificar (MR) la clave primaria si no existen claves ajenas que la referencien. Es decir, si esa tupla no está refernciada en otra tabla. Por ejemplo, sólo podremos modificar la clave primaria de una determinada tupla, si su valor no actúa de clave ajena en ninguna otra tabla. (En los ejemplos, solo podríamos borrar la fila que contiene los datos de HOLANDA, porque no existe ninguna referencia a este país en la tabla de JUGADORES).
- **transmisión en cascada (CASCADE)**: el Borrado (BC) o la modificación (MC) de tuplas de R1 lleva consigo el BC/MC en cascada de las tuplas de R2. (Mirando al ejemplo, si borramos la fila de INGLATERRA en la tabla de PAIS, tendríamos que eliminar todos las filas de la tabla JUGADOR, que contengan a un jugador INGLÉS, las tablas del ejemplo quedarían del siguiente modo:

Tabla (Relación): PAISES					**Tabla (Relación): JUGADORES**		
Id_Pais	**Nombre**	**Habitantes**	**Idioma**		**Id_Jugador**	**Nombre**	**Id_Pais**
ESP	ESPAÑA	9.000.000	ESPAÑOL		J001	Raul Lopez	ESP
					J002	Robinho	BRA
FRA	FRANCIA	11.345.655	FRANCÉS		J003	Zinedine Zidane	FRA
BRA	BRASIL	23.098.476	PORTUGUÉS				
HOL	HOLANDA	5.000.765	HOLANDÉS				

- **puesta a nulos (SET NULL):** el B/M de tuplas de R1 pone a nulo los valores de las tuplas de R2. (En el ejemplo, si borrásemos el país INGLATERRA, la tas tablas quedarían del siguiente modo:

Tabla (Relación): PAISES					**Tabla (Relación): JUGADORES**		
Id_Pais	**Nombre**	**Habitantes**	**Idioma**		**Id_Jugador**	**Nombre**	**Id_Pais**
ESP	ESPAÑA	9.000.000	ESPAÑOL		J001	Raul Lopez	ESP
					J002	Robinho	BRA
FRA	FRANCIA	11.345.655	FRANCÉS		J003	Zinedine Zidane	FRA
BRA	BRASIL	23.098.476	PORTUGUÉS		J004	Beckam	**NULL**
HOL	HOLANDA	5.000.765	HOLANDÉS				

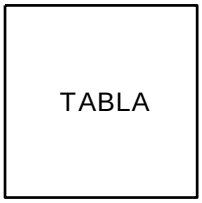
- **puesta a valor por defecto (SET DEFAULT):** el B/M de tuplas de R1 pone a un valor por defecto definido los valores de la clave ajena R2. (En las tablas del ejemplo, si consideramos que eliminamos el pais INGLATERRA, la tabla quedaría del siguiente modo, considerando que tomamos como valor por defecto NGN, que significa ninguno:

Tabla (Relación): PAISES					Tabla (Relación): JUGADORES		
Id_Pais	**Nombre**	**Habitantes**	**Idioma**		**Id_Jugador**	**Nombre**	**Id_Pais**
ESP	ESPAÑA	9.000.000	ESPAÑOL		J001	Raul Lopez	ESP
					J002	Robinho	BRA
FRA	FRANCIA	11.345.655	FRANCÉS		J003	Zinedine Zidane	FRA
BRA	BRASIL	23.098.476	PORTUGUÉS		J004	Beckam	**NGN**
HOL	HOLANDA	5.000.765	HOLANDÉS				

3.5 Técnica: Diagrama de Estructura de Datos (DED)

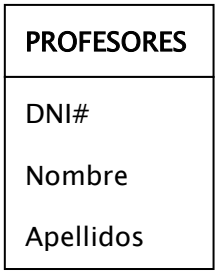
La técnica empleada en la fase de diseño del modelo de datos, es el Diagrama de Estructura de datos. Éste sirve tanto para representar el modelo relacional (visto en este tema) como para los modelos en red y jerárquico (que no veremos en este curso).

3.5.1 Tablas ó relaciones



3.5.2 Campos o atributos

Los campos (atributos) de la tabla se pueden indicar dentro de la tabla. El nombre de la tabla se separa de los campos por una línea.



3.5.3 Interrelaciones ó tablas

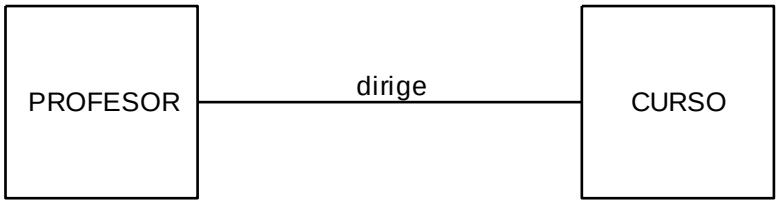
Existen unas reglas que deben de cumplir las interrelaciones son:

→ Solo se permiten interrelaciones binarias (entre 2 tablas) o reflexivas.

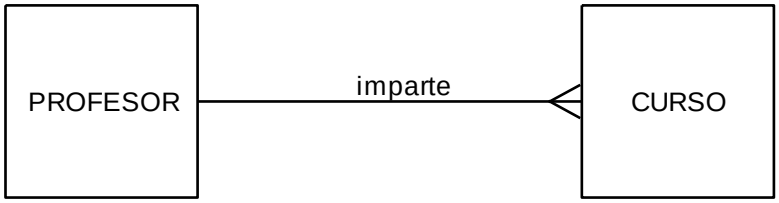
→ No existen relaciones N:M, solo 1:1 y 1:N.

Las relaciones se representan mediante una línea que une las 2 tablas. Para representar la parte N (muchos) se utiliza un tridente.

RELACIÓN 1:1

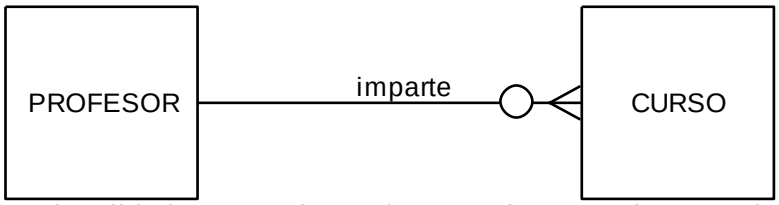


RELACION 1:N



OPCIONALIDAD

Para representar la opcionalidad (que la relación es opcional en un extremo de la misma) se utiliza un circulo.



Opcionalidad en curso (un profesor puede que no imparta ningún

3.6 Transformación del modelo E/R al modelo lógico relacional

El paso de E-R a modelo lógico relacional produce cierta pérdida de semántica ya que las entidades y las interrelaciones se convierten en tablas, pero la pérdida de semántica no implica, sin embargo, un peligro para la integridad de la base de datos, ya que se definen las restricciones de integridad referencial que aseguran la conservación de la misma.

3.6.1 Tres reglas básicas en las transformaciones del modelo Entidad-Relación a Relacional

Existen tres reglas básicas que se deben cumplir en la transformación del modelo entidad relación al modelo relacional, reglas que se explicarán más detalladamente en los apartados que siguen a éste, son:

→ Toda entidad se convierte en una relación (o tabla).

→ Toda interrelación N:M se transforma en una relación (también llamada tabla).

→ Toda interrelación 1:N se traduce en el fenómeno "propagación de claves".

3.6.2 TRANSFORMACIÓN DE ENTIDADES

→ Cada entidad del esquema E/R dará lugar a una nueva ***relación ó tabla*** cuya clave primaria es el identificador principal de la entidad

→ Cada atributo de una entidad se transforma en un atributo de la relación creada para la entidad aunque hay que tener en cuenta sus distintos tipos de restricciones semánticas

- **Atributos univaluados**: dan lugar a un atributo de la relación
- **Atributos multivaluados**: dan lugar a una nueva relación (tabla) 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
- **Atributos obligatorios**: atributos con la restricción de NOT NULL
- **Atributos opcionales**: atributos que pueden tomar valores NULL
- **Identificador principal**: atributos que forma la clave primaria
- **Identificador alternativo**: atributos con la restricción de UNIQUE

3.6.3 TRANSFORMACIÓN DE INTERRELACIONES BINARIAS

3.6.3.1 INTERRELACIONES 1:1

Dependiendo de la cardinalidad de las interrelaciones, la transformación se realiza:

→ **Norma general**: la clave primaria de una tabla se introduce como clave ajena de la otra tabla. Los atributos de la relación se pasan a la tabla donde se haya introducido la clave ajena. Los atributos de la relación se almacenan siempre donde se almacena la relación, es decir en la tabla en donde se define nueva columna que es clave ajena.

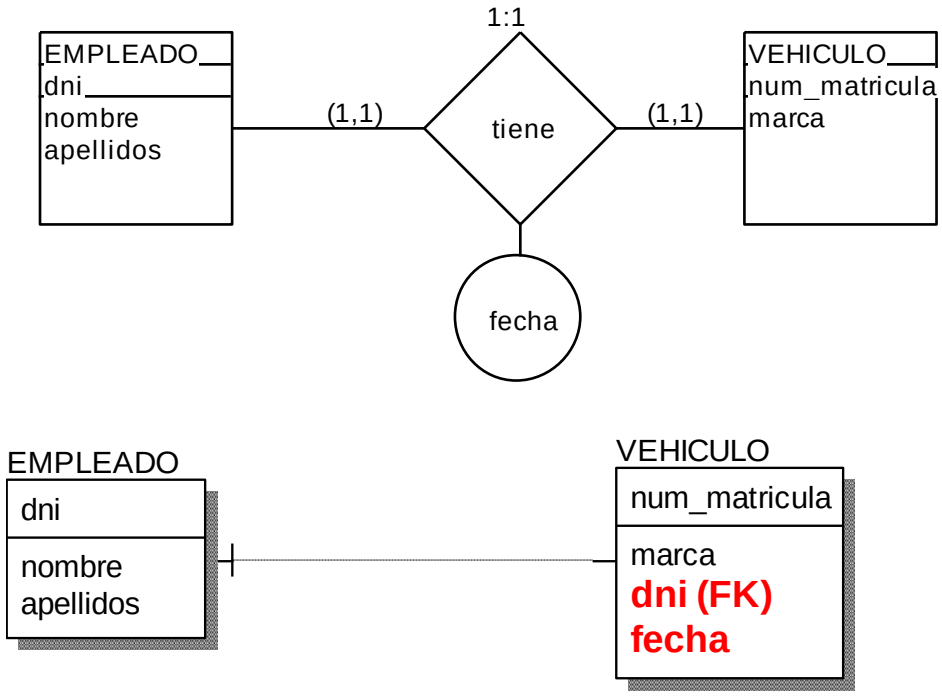


Figura 5. Transformación de interrelación 1:1

→ **(1,1) (1,1)**: se puede seguir la norma general en cualquier sentido. La relación se almacena en cualquiera de las dos tablas mediante la definición de una FK. También se podría crear una nueva tabla para la interrelación. Restricción NOT NULL en clave ajena, es decir, la clave ajena no puede tener un valor nulo.

→ **(1,1) (0,1)**: se sigue la norma general creando la clave ajena en la tabla de la entidad que participa con (1,1), es decir, se propaga la clave de la relación con cardinalidad (0,1) a la relación con cardinalidad (1,1). Para evitar valores null en la BD. Se añade restricción NOT NULL en clave ajena.

→ **(0,1) (0,1)**. La relación se almacena en cualquiera de las dos tablas mediante la definición de una FK. También se podría crear una nueva tabla para la interrelación, especialmente si la cardinalidad de las tablas es muy alta. Para crear la clave ajena lo haríamos en la tabla de menor cardinalidad. Restricción NULL en clave ajena, es decir, la clave ajena puede tener valores nulos.

→

→ INTERRELACIÓN 1:N

→ **(1,1) (x,N):** se propaga la clave primaria a la tabla de la entidad que particia con 1, junto con los atributos de la relación. Se añade restricción NOT NULL en clave ajena.

Ejemplo, siguiendo el diagrama entidad-relación de la Figura 4. Ejemplo de Entidad Relación (JUGADORES DE FUTBOL), el modelo relacional quedaría del siguiente modo:

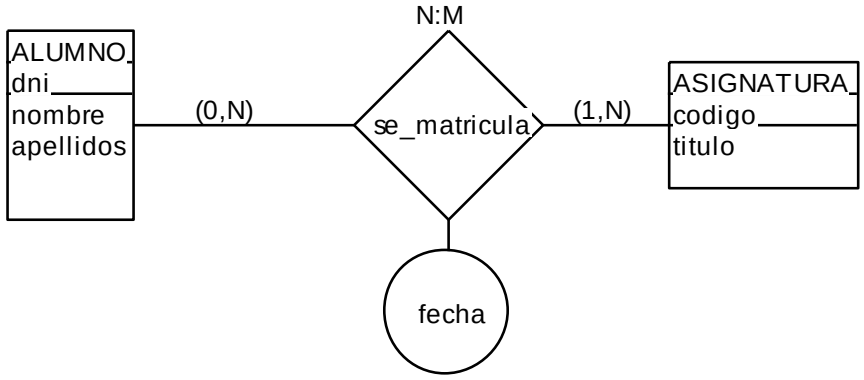
JUGADOR(Id_jugador#, Nombre, Id_pais)

PAIS (Id_pais#, Nombre, Habitantes, Idioma)

3.6.3.2 **(0,1) (X,N):** se propaga la clave primaria a la tabla de la entidad que particia con 1, junto con los atributos de la relación. Se añade restricción NULL en clave ajena, la clave ajena podrá tomar valores nulos.

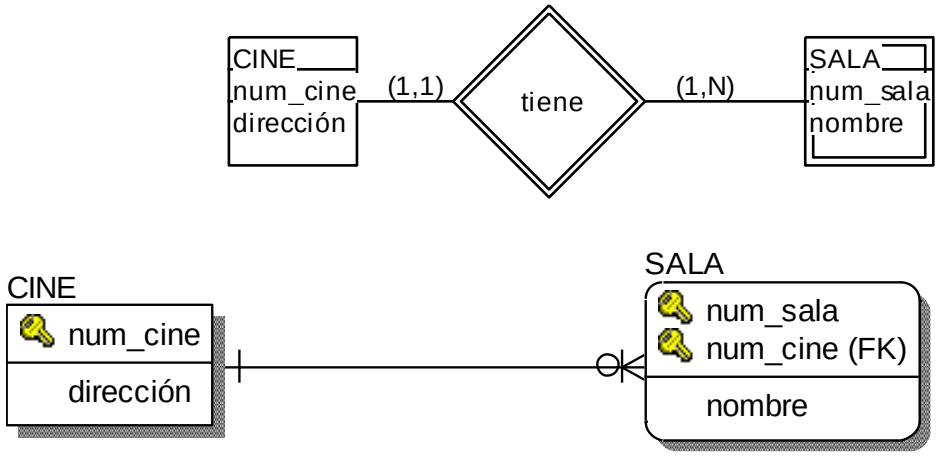
3.6.3.3 INTERRELACIÓN N:M

→ Siempre dan lugar a NUEVA TABLA o RELACIÓN cuya clave será la concatenación de los Identificadores principales de las Entidades que enlaza la interrelación. De esta forma los atributos que forma la clave primaria en esta nueva relación son claves ajenas respecto a las relaciones en las que estos atributos son clave primaria



3.6.3.4 DEPENDENCIA EN EXISTENCIA E IDENTIFICACION

Suelen ser siempre 1:N por lo que no generan tabla. La clave de la entidad fuerte debe introducirse en la tabla de la entidad débil **y formar parte de esta** (además de ser clave ajena). La dependencia en existencia siempre se traduce en un Borrado en Cascada.



3.6.3.5 Dependencia de Existencia

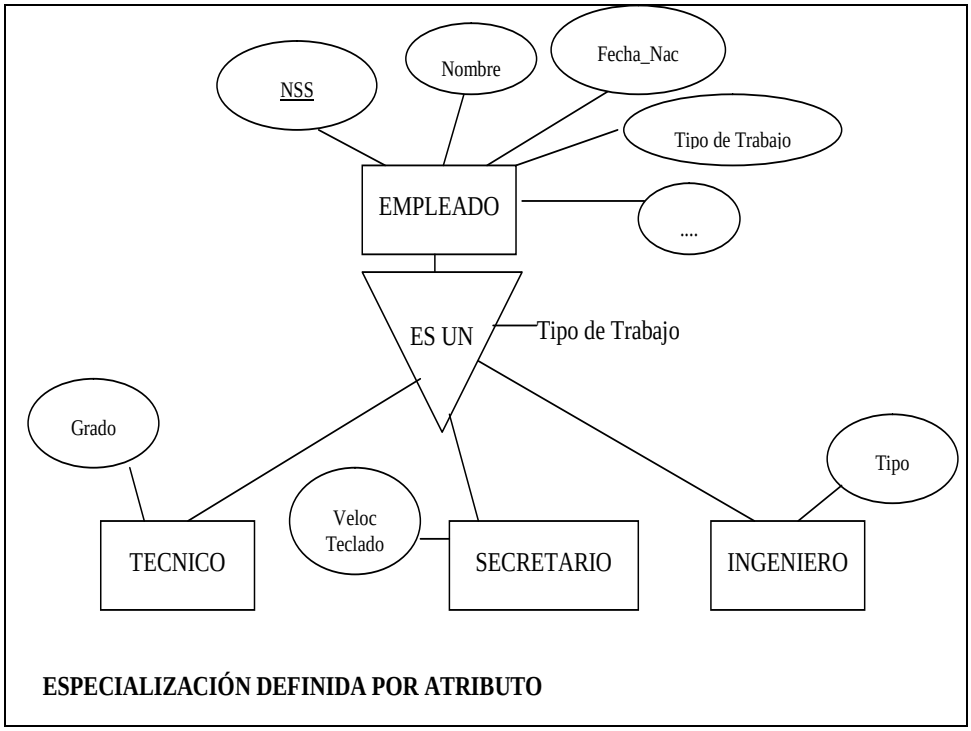
Una interrelación 1:N de Dependencia en Existencia origina que la clave ajena propagada desde la entidad fuerte a la entidad débil. Además debe tener la opción de DELETE ON CASCADE, es decir, no puede existir ninguna ocurrencia de la entidad hijo que no esté relacionada con una ocurrencia de la entidad padre.

3.6.4 TRANSFORMACIÓN DE RELACIONES TERNARIAS

Se crea una nueva tabla para la relación. Esta tabla tendrá como claves ajenas las claves primarias de las relaciones que relaciona, y su clave primaria estará formada por las claves ajenas de aquellas tablas en las que su cardinalidad es (x, N).

3.6.5 TRANSFORMACIÓN DE RELACIONES JERARQUICAS

Suponiendo el siguiente diagrama ER, para convertir cada superclase C con m subclases {s1,s2,..,Sn} a relaciones …



… podríamos seguir una de las siguientes opciones:

1) **Crear una relación para la superclase C, siguiendo las reglas de un tipo de entidad, y crear tantas relaciones como subclases haya que contengan**:
   - Los atributos propios de la subclase
   - Los atributos que conformen la clave primaria de la superclase, que constituirán la clave primaria de la relación

   EMPLEADO (**NSS_EMP**, NOMBRE_EMP, FNAC_EMP, TIPOTRABAJO)

   SECRETARIO (***NSS EMP***,VELO_SEC)

   TECNICO (***NSS EMP***,GRADO_TEC)

   INGENIERO (***NSS EMP***,TIPO_ING)

2) **No crear una relación para la superclase y crear tantas relaciones como subclases haya que contengan**:
   - Los atributos propios de la subclase que no formen parte de la clave primaria
   - Los atributos propios de la superclase que no formen parte de la clave primaria
   - Los atributos que conformen la clave primaria de la superclase, que constituirán la clave primaria de la relación

   SECRETARIO (**NSS SEC**, NOMBRE_SEC, ,FNAC_SEC, TIPOTRABAJO, VELO_SEC)
   TECNICO (**NSS EMP**, NOMBRE_TEC, FNAC_TEC, TIPOTRABAJO, GRADO_TEC)
   INGENIERO (**NSS EMP**, NOMBRE_ING,APE1,FNAC_ING, TIPOTRABAJO,TIPO_ING)

Esta opción **sólo es válida** si las subclases son **disjuntas y total**. Sí es solapada una entidad que pertenezca a más de una subclase tendrá los atributos heredados de la superclase almacenados en más de una relación. Si no es total una entidad que no pertenezca a ninguna de las subclases se perdería.

3) **Crear una única relación que contenga**:

- Los atributos propios de la superclase que no formen parte de la clave primaria
- Los atributos propios de cada subclase que no formen parte de la clave primaria
- Un atributo t que indica a que subclase pertenece cada tupla
- Los atributos que conformen la clave primaria de la superclase, que constituirán la clave primaria de la relación

Esta opción es válida cuando las subclases son disjuntas y existe un atributo que indica la subclase a la cual cada tupla pertenece. Esta opción tiene un potencial alto de generar valores nulos. Si la especialización es parcial t puede tener valor nulo y en consecuencia todos los atributos de todas las subclases tendrán valores nulos. Si la especialización es definida por atributo no es necesario el atributo t.

Si hay muchos atributos específicos puede no ser recomendable.

EMPLEADO (**NSS_EMP**, NOMBRE_EMP, FNAC_EMP, TIPOTRABAJO, VELO_SEC, GRADO_TEC, TIPO_ING)

4 Normalización

Una vez obtenido el modelo lógico hay que comprobar que está normalizado. El modelo resultante se llama modelo lógico normalizado.

4.1 Dependencia funcional

Un atributo Y depende funcionalmente de otro X, si y sólo sí a cada valor de X le corresponde un único valor de Y. También se dice que X determina Y (X recibe el nombre de **determinante**). $X \rightarrow Y$

4.2 Dependencia funcional completa

Un atributo Y tiene dependencia funcional completa de otro X, si depende de él en su totalidad, es decir, no depende de ninguno de los posibles atributos que formen parte de X.

$X \Rightarrow Y$

código, nombre $\rightarrow$ teléfono

código, nombre =/=> teléfono YA QUE código $\rightarrow$ teléfono

4.3 Dependencia funcional transitiva

Un atributo depende transitivamente de otro si y sólo sí depende de él a través de otro atributo. Así Z depende transitivamente de X, sí:

$X \rightarrow Y$

Y -/-> X

$Y \rightarrow Z$

4.4 Primera forma normal (1FN)

Una tabla está en 1FN si no tiene grupos repetitivos, es decir, un atributo solo puede tener un único valor.

cod-libro	titulo	cod-autor
001	ADAIG	035
002	PLE	045
		055

NO ESTARIA EN 1FN, porque en cod-autor aparecen dos valores para codigo autor 045 y 055, estaría en 1FN si en cod-autor solo apareciese 045 y 055.

4.5 Segunda forma normal (2FN)

Una tabla está en 2FN sí está en 1FN y además todos los atributos que no forman parte de la clave tienen dependencia funcional completa de la clave.

LIBRO (cod-libro, título, cod-autor)

cod-libro → título (por tanto NO ESTÁ EN 2FN).para que esté en 2FN hay que dividir la relación:

LIBRO (cod-libro, título)
AUTOR (cod-libro, cod-autor)

4.6 Tercera forma normal (3FN)

Una tabla está en 3FN si se encuentra en 2FN y no existen dependencias funcionales transitivas de atributos no principales respecto de la clave.

EMPLEADO (cod-empl, cod-dpto, nombre-dpto)

cod-empl → cod-dpto
cod-dpto -/-> cod-empl
cod-dpto → nombre-dpto

NO está en 3FN. solución:

EMPLEADO (cod-empl, cod-dpto)
DEPARTAMENTO (cod-dpto, nombre-dpto)

4.7 Forma normal de Boyce-Codd (FNBC)

Una tabla está en FNBC sí y sólo sí todo **determinante** es clave candidata.

5 DOCUMENTACIÓN

La metodología de la administración española es Métrica v3. En esta metodología la fase de análisis se denomina "ASI = Análisis de sistemas de información", en esta fase se indica que:

1) Se debe **realizar la especificación de requisitos** (que se comenzó en la fase anterior EVS). En la especificación de requisitos se recogen todos los requisitos definidos por el usuario y que serán necesarios para poder realizar el análisis conceptual de datos.
2) Después se tiene que realizar el **modelo conceptual de datos**: se parte de la especificación de requisitos (donde los datos que se necesitan guardar están expresados en lenguaje natural) y se genera el modelo conceptual de datos utilizando alguna TÉCNICA descriptiva como pude ser el Diagrama de E/R.
3) Después se tiene que obtener el **modelo lógico de datos** a partir del modelo conceptual de datos.
4) Y se debe aplicar las normas de normalización para obtener el modelo lógico de datos normalizado.

Además de incluir el diagrama de DED del modelo lógico de datos normalizado en la fase de ASI, toda la información sobre el modelo lógico de datos queda recogida en el diccionario de datos:

5.1 Diccionario de datos

En el diccionario de datos se recoge la siguiente información sobre el modelo conceptual de datos:

→ Nombre de Las entidades y descripción

→ Nombre de las relaciones y descripción

Atributos de las entidades, descripción de cada atributo y definición (tipo de dato, longitud, restricciones, ...)

Se utiliza la Notación **BNF**

=	**Compuesto de**
+	y
( )	Opcionalidad
{ }	Iteración
[|]	Selección de alternativas
*** ***	Comentario
@	Clave

Se pueden indicar los límites de las iteraciones

Palabra = {Letra}	**Sin límite (indeterminado)**

Palabra = 1{Letra}	Mínimo 1, máximo indeterminado
Palabra = {Letra}10	Mínimo indeterminado, máximo 10
Palabra = 1{Letra}10	Mínimo 1, máximo 10
Palabra = 10{Letra}10	Exactamente 10

Ejemplos:

```
Alumno =  Nombre + Apellido
Alumno = { Carácter }
Apellido = { Carácter }
Carácter = [ "A"-"Z" | "a"-"z" ]
```