TEMA 2: DISEÑO DE BASE DE DATOS

2.1 DIAGRAMA ENTIDAD / RELACION

Bajo la estructura de la base de datos se encuentra el modelo de datos: una colección de herramientas conceptuales para describir los datos, las relaciones, la semántica y las restricciones de consistencia. El modelo de datos entidad-relación (E-R) está basado en una percepción del mundo real que consta de una colección de objetos básicos, llamados entidades, y de relaciones entre estos objetos. Una entidad es una «cosa» u «objeto» en el mundo real que es distinguible de otros objetos. Por ejemplo, cada persona es una entidad, y las cuentas bancarias pueden ser consideradas entidades.

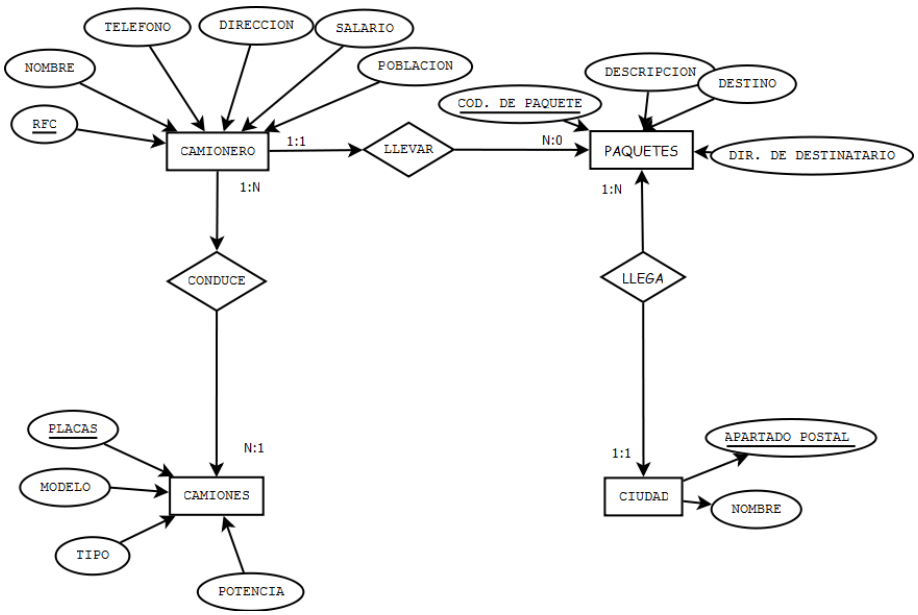
Las entidades se describen en una base de datos mediante un conjunto de atributos. Por ejemplo, los atributos número-cuenta y saldo describen una cuenta particular de un banco y pueden ser atributos del conjunto de entidades cuenta. Análogamente, los atributos nombre-cliente, calle-cliente y ciudad-cliente pueden describir una entidad cliente.

Un atributo extra, id-cliente, se usa para identificar unívocamente a los clientes (dado que puede ser posible que haya dos clientes con el mismo nombre, el diseño de la base de datos en el nivel lógico. Una base de datos puede tener también varios esquemas en el nivel de vistas, a menudo denominados subesquemas, que describen diferentes vistas de la base de datos.

Una relación es una asociación entre varias entidades. Por ejemplo, una relación impositor asocia un cliente con cada cuenta que tiene. El conjunto de todas las entidades del mismo tipo, y el conjunto de todas las relaciones del mismo tipo, se denominan respectivamente conjunto de entidades y conjunto de relaciones.

La estructura lógica general de una base de datos se puede expresar gráficamente mediante un diagrama ER, que consta de los siguientes componentes:

- Rectángulos, que representan conjuntos de entidades.
- Elipses, que representan atributos.
- Rombos, que representan relaciones entre conjuntos de entidades.
- Líneas, que unen los atributos con los conjuntos de entidades y los conjuntos de entidades con las relaciones.



2.1.1 Entidades

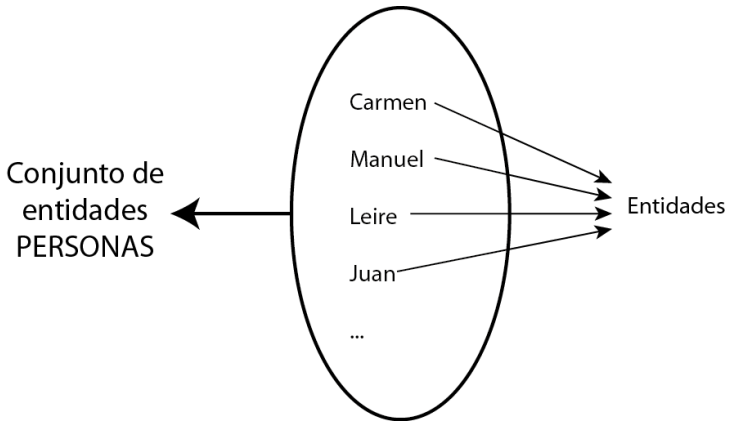
Se trata de cualquier objeto u elemento (real o abstracto) acerca del cual se pueda almacenar información en la base de datos. Es decir cualquier elemento informativo que tenga importancia para una base de datos.

Ejemplos de entidades son: una persona que se llama Pedro, la factura número 32456, el coche matrícula 3452BCW, etc. Una entidad no es un propiedad concreta, sino un objeto que puede poseer múltiples propiedades (atributos). Es decir "Sánchez" es el contenido del atributo Primer Apellido de la entidad que representa a la persona Pedro Sánchez Crespo con DNI 12766374,...

Las entidades son objetos completos, con todos los valores de las propiedades de dicho objeto. Descubrir entidades es la tarea principal del diseño de esquemas Entidad/Relación.

Las entidades que poseen las mismas propiedades, porque se refieren al mismo tipo de ente, forman conjuntos de entidades. En la actualidad se suele llamar entidad a lo que anteriormente se ha definido como conjunto de entidades, seguramente por una cuestión de ahorro en el lenguaje. De este modo hablaríamos de la entidad PERSONAS. Mientras que cada persona en concreto sería una ocurrencia o un ejemplar de la entidad persona.

- Cada conjunto de entidades tiene una llave
- Cada atributo tiene un dominio



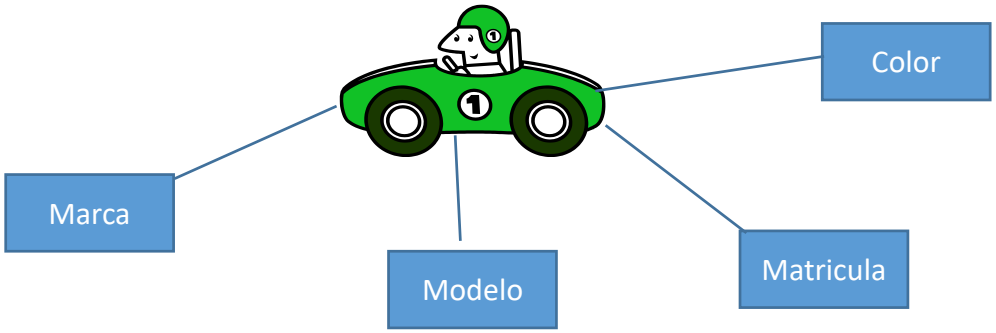
TIPOS DE ENTIDADES

- **Entidades fuertes:** es aquella cuya existencia es absoluta.
- **Entidades débiles:** es aquella cuya existencia depende de otra (considerada su entidad fuerte). Se trata de entidades totalmente supeditadas a otras, de modo que si un ejemplar de la entidad fuerte desaparece, todos los ejemplares de la entidad débil relacionados, desparecerán también del sistema. Las entidades débiles ocurren cuando hay una entidad más fuerte de la que dependen, en el sentido de que la propia existencia de la entidad débil está supeditada a la existencia de su entidad fuerte. Lógicamente tienen relación con esa entidad, y es esa relación es la que marca el hecho de que una entidad es débil y la otra fuerte. Un ejemplo de entidad débil es una relación entre autores y libros, en este caso los autores sería una entidad fuerte mientras que los libros sería una entidad débil ya que si no existiese la entidad autores, la entidad libros tampoco existirían.

2.1.2 Atributos

Describen propiedades de las entidades y las relaciones. Son fundamentales y establecen la información que deseamos almacenar de cada objeto de la base de datos.

- Permiten representar una entidad utilizando un conjunto de atributos.
- Describen cada entidad por medio de un par valor-atributo.



Los atributos se pueden clasificar:

- **Simples o compuestos**
  - **Simple:** no se puede dividir en otros atributos. Ej → Segundo apellido
  - **Compuesto:** puede dividirse en otros atributos. Ej → Nombre y apellidos.
- **Valor único o multivalores**
  - **Valor único:** sólo un valor para cada entidad. Ej → fecha nacimiento.
  - **Multivalores:** un atributo puede tener varios valores. Ej → Número de teléfono.
- **Atributos derivados**
  - Valor calculado a partir de otra información ya existente (atributos, entidades relacionadas)
  - Son información redundante. → Edad de un empleado cálculo a partir de fechanacimiento
- **Atributos NULL** u opcionales
  - No es un valor como tal, es un estado. Significa valor desconocido.
  - En realidad no es un atributo como tal sino un valor que puede adquirir.
    - El valor es desconocido bien porque falta o porque no se puede conocer.

CLAVES O LLAVES

Dentro de una entidad tiene que haber un atributo principal que identifica a la entidad y su valor tiene que ser único. Se trata de uno o más atributos de una entidad cuyos valores son únicos en cada ejemplar de la entidad. Es decir todos los elementos de una entidad tienen en ese (o esos) atributo, un valor diferente (y nunca vacío).

Este tipo de atributos son fundamentales y se marcan en el esquema subrayando el nombre del identificador. Para que un atributo sea considerado un buen identificador tiene que cumplir con los siguientes requisitos:

- Deben distinguir a cada ejemplar de la entidad o relación. Es decir no puede haber dos ejemplares con el mismo valor en el identificador.
- Todos los ejemplares de una entidad deben tener el mismo identificador.

- Un identificador puede estar formado por más de un atributo.
- Puede haber varios identificadores candidatos, en ese caso hay que elegir el que tenga más importancia en nuestro sistema (el resto pasan a ser alternativos).

Todas las entidades deben de tener un identificador, en el caso de que una entidad no disponga de un identificador (puede ocurrir, pero hay que ser cauteloso, a veces se trata de entidades que están mal modeladas) entonces hay que añadir un atributo que haga de identificador. Los futuros valores de este atributo no nos interesan, lo único que necesitamos de él es que disponga de un valor diferente para cada ejemplar de la entidad. El nombre de este atributo artificial es la palabra id seguida del nombre de la entidad.

Existen dos tipos de claves:

- Superllave: Uno o más atributos que nos permite identificar una entidad en específico dentro de un conjunto de entidades y ninguna otra entidad la tiene. Toda relación tiene por lo menos una superllave llamada llave primaria.
    - Clave primaria (Primary Key): es el valor o conjunto de valores que identifican una fila dentro de una tabla. Nunca puede ser NULL. Un ejemplo claro de clave primaria seria el DNI, que es único para cada persona y no puede ser NULL.
    - Clave ajena (Foreign Key): es el valor o valores de una tabla que corresponde con el valor de una clave primaria en otra tabla. Esta clave es la que representa las relaciones entre las tablas.
- Llave candidata: Son aquellos atributos que tienen características para ser superllaves, pero hay dos o más en una entidad; una se tomara como llave primaria y otra como llave secundaria.

A la hora de elegir un identificador principal debemos tener presente los siguientes principios ya que la elección de un buen identificador simplifica el diseño de la base.

- Siempre debemos elegir el candidato que tenga más que ver con el problema que estamos resolviendo.
- Sólo debemos elegir como identificadores principales, a atributos (sea uno o varios cuyos valores son únicos) a fechas, números enteros y textos cortos y de tamaño fijo.
- De los posibles candidatos observados (que cumplan el punto anterior), elegir el que tenga más relación con el problema que estamos resolviendo. Si no encontramos una diferencia conceptual, entonces elegir el que tenga un tamaño más corto.
- Si ningún candidato cumple estas reglas, es mejor incluso inventarse un identificador (al final contendrá un número entero diferente para cada elemento de la entidad).

2.1.3 Relaciones

Representan asociaciones entre entidades, es decir entidades de un conjunto que tienen contacto con entidades de otro conjunto. Es el elemento del modelo que permite relacionar en sí los datos del mismo;

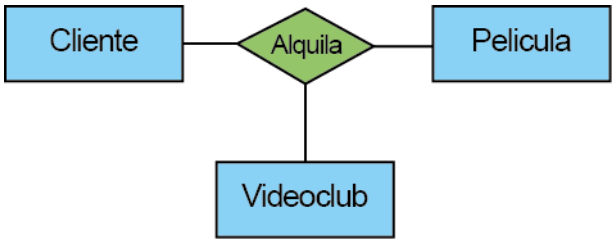
de otro modo tendríamos información aislada. Por ejemplo, en el caso de que tengamos una entidad personas y otra entidad trabajos. Ambas se relacionan ya que las personas trabajan y los trabajos son realizados por personas.

Una regla fundamental es que en una relación no pueden aparecer dos veces relacionados los mismos ejemplares de cada entidad. Una relación estable una asociación entre un ejemplar de una entidad con un ejemplar de otra entidad, una vez. Es decir; en el ejemplo anterior, en la relación no puede aparecer dos veces el mismo trabajador asociado al mismo trabajo.

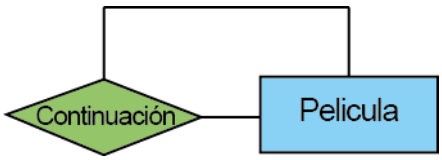
Los tipos de relaciones que se pueden dar entre varias entidades son:

- **Relaciones binarias:** son las relaciones típicas. Se trata de relaciones que asocian dos entidades.

- **Relaciones ternarias:** relacionan tres entidades. A veces se pueden simplificar en relaciones binarias, pero no siempre es posible.

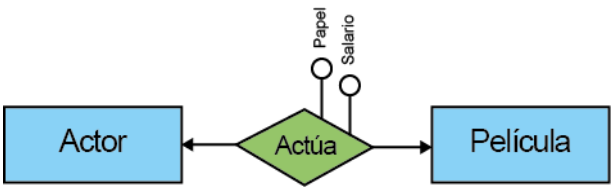


- **Relaciones reflexivas o recursivas:** es una relación que sirve para relacionar dos ejemplares de la misma entidad (personas con personas, piezas con piezas, etc.)



Siempre que se pueda las relaciones utilizadas han de ser binarias. Esto es posible cuando dos relaciones binarias mantienen la misma semántica que la ternaria. Las relaciones ternarias (o en general n-arias) deben ser usadas cuando se desea evidenciar la participación de las tres entidades (o en general de las n entidades).

Una relación puede tener atributos descriptivos.



RELACIONES EN ENTIDADES DÉBILES

A la hora de establecer las relaciones hay que indicar en primer término el número de relaciones en las que una entidad puede aparecer. Se anota en términos de:

• **Cardinalidad mínima**. Indica el número mínimo de asociaciones en las que aparecerá cada ejemplar de la entidad (el valor que se anota es de cero o uno, aunque tenga una cardinalidad mínima de más de uno, se indica sólo un uno)

• **Cardinalidad máxima**. Indica el número máximo de relaciones en las que puede aparecer cada ejemplar de la entidad. Puede ser uno, otro valor concreto mayor que uno (tres por ejemplo) o muchos (se representa con n). Normalmente la cardinalidad máxima es 1 ó n.

En los esquemas entidad / relación la cardinalidad se puede indicar de muchas formas. Quizá la más completa consiste en anotar en los extremos la cardinalidad máxima y mínima de cada entidad en la relación.

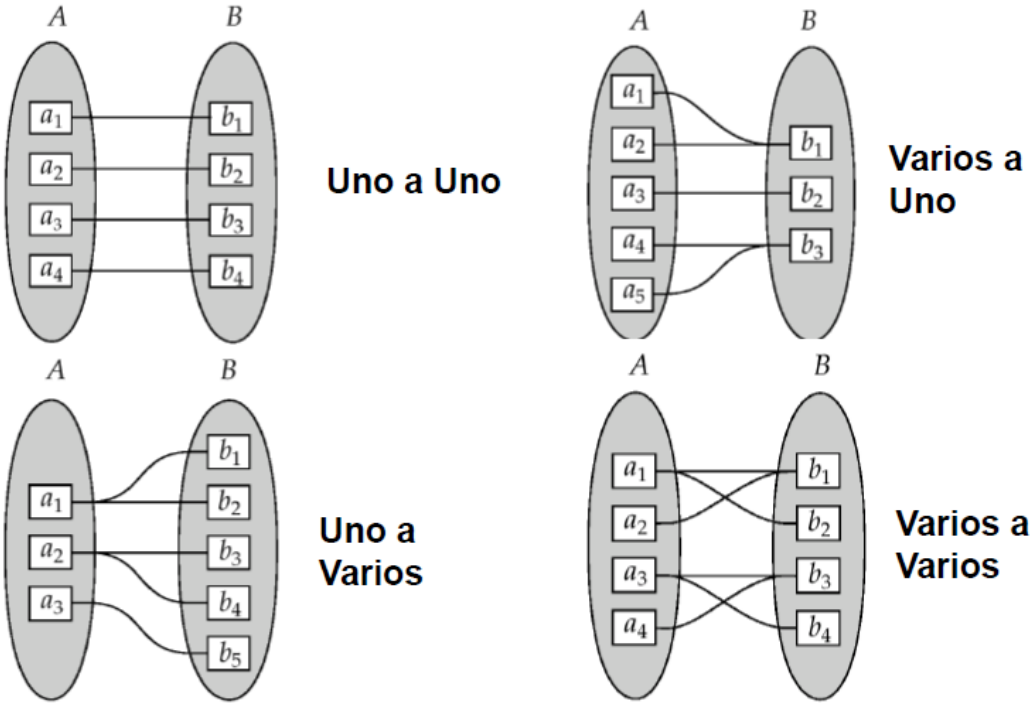
En el ejemplo un jugador tiene una cardinalidad mínima de 0 (puede no estar en ningún equipo) y una máxima de 1 (como mucho está en un equipo, no puede estar en dos a la vez). Cada equipo tiene una cardinalidad mínima de uno (en realidad sería una cardinalidad mínima más alta, pero se anota un uno) y una máxima de n (en cada equipo hay muchos jugadores)

Una participación de una entidad en una relación puede ser:

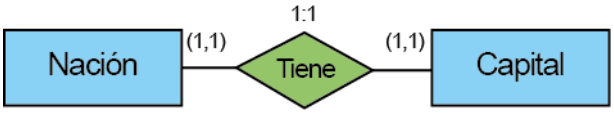
• **Total:** Todas las entidades participan en al menos una relación.

• **Parcial:** Algunas de las entidades pueden no participar en la relación.

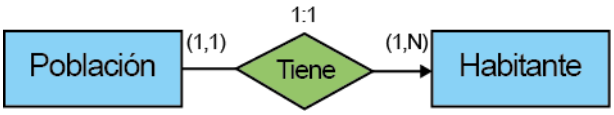
Para expresar el número de entidades que están asociadas a otra mediante un conjunto de relaciones.



• **Una a una (1,1).** Una entidad en A se asocia como mucho con una entidad en B y una entidad en B sólo puede relacionarse con una de A.



- **Una a muchos (1, n).** Una entidad en A se puede asociar con cualquier número (cero a más) de entidades en B; sin embargo, B solo puede asociarse con una entidad de A



- **Muchos a uno (n,1).** Una entidad en A solo puede asociarse con una entidad de B, pero una entidad de B puede asociarse con muchas entidades de A.

- **Muchos a muchos**. Una entidad de A puede asociarse con muchas entidades de B y B puede asociarse con muchas entidades de A.

RELACIONES EN ENTIDADES DÉBILES

- **Dependencia en existencia: h**ay dependencia de existencia cuando en un tipo de relación se está vinculando una entidad regular (ER) con una entidad débil (ED), esto de tal forma que la ocurrencia de la entidad débil no exista sin la ocurrencia de la entidad regular de la cual depende la ED.

Por ejemplo en una base en la que relacionamos autores y libres, la entidad autor, la relación escribe y la entidad débil libros. Esto sería “El autor escribe libros y varios libros son escritos por un autor”. Si los Autores no existirán tampoco los libros ya que dependen de la escritura de un autor. En la relación que se representa por un rombo se pone una E en la parte superior para especificar la existencia.

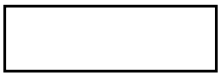
- **Dependencia en identificación:** hay dependencia de identificación es cuando además de la dependencia de existencia hay una entidad débil pero estas no se pueden identificar nada más por sus atributos si no que tiene que agregar la clave de existencia de la entidad regular de la cual depende.

Por ejemplo en una base en la que relacionamos libros ejemplares tenemos que los libros tienen un código de libro y los ejemplares No. de ejemplar. Esto los hace diferentes.

2.1.4 Símbolos

La simbología del diagrama entidad relación que se utiliza es:

- Rectángulo simple: conjunto de entidades fuerte.



- Rectángulo doble: conjunto de entidades débil.

- Elipse simple: atributo

- Elipse rellena: clave primaria

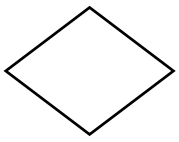
- Elipse medio rellena: clave candidata

- Elipse doble: atributo compuesto

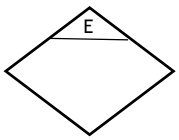
- Elipse con flecha: atributo multivalorado

- Elipse discontinua: atributo opcional

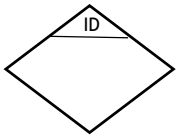
- Rombo: conjunto de relaciones



- Rombo E: relación con dependencia en existencia.

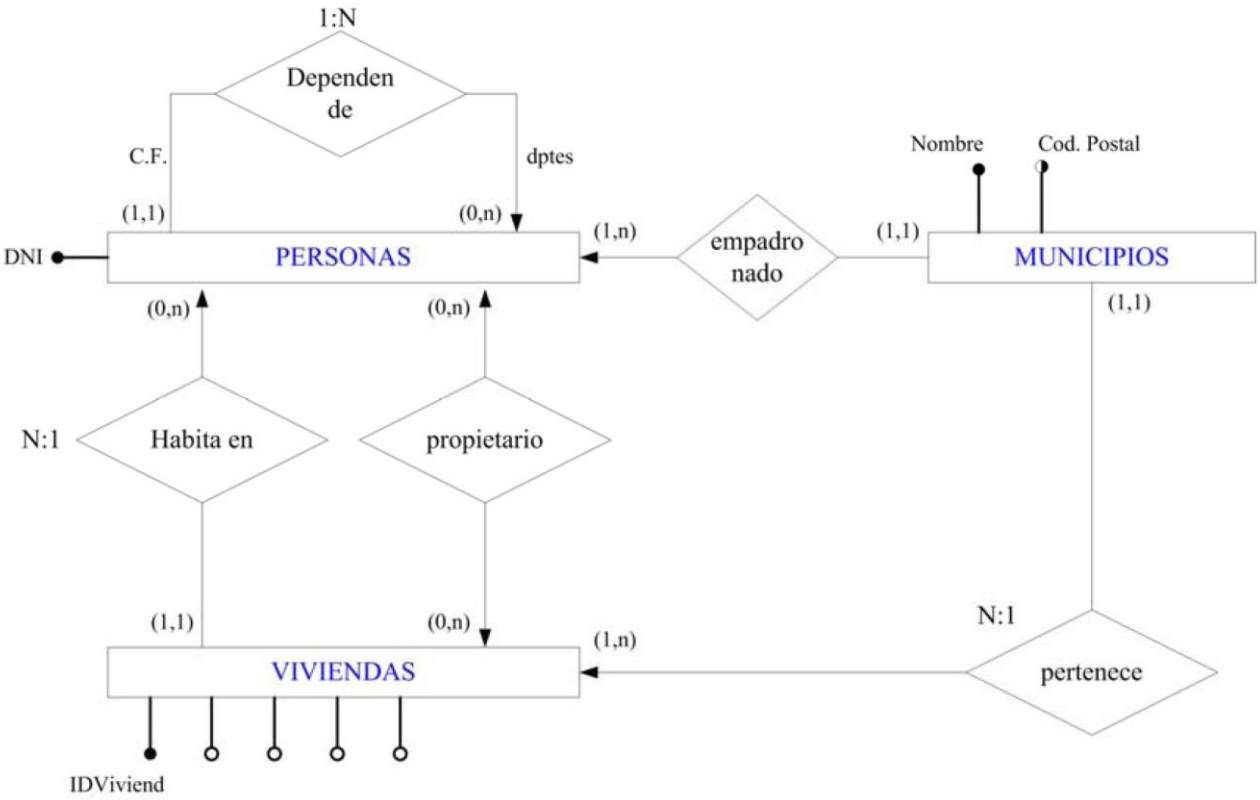


- Rombo ID: relación con dependencia en identificación.



- Líneas simples: unen atributos con los conjuntos de entidades, atributos con los conjuntos de atributos o conjuntos de entidades con conjuntos de relaciones.
    - Líneas indirectas: cardinalidad 1
    - Con flecha: cardinalidad N

Ejemplo de diagrama entidad relación.



2.2 DIAGRAMA ENTIDAD / RELACION EXTENDIDO

Dentro del diagrama entidad relación hemos visto únicamente como crear los conceptos básicos de un diagrama entidad relación. Existe una serie de relaciones que se pueden incluir en el diagrama que constituyen el diagrama relación extendido en el cual se incluyen posibilidades de herencia, agregación y otro tipo de entidades.

2.2.1 Relaciones ISA (ES A)

Son relaciones que indican relaciones que permiten distinguir tipos de entidades, es decir tendremos entidades que son un (is a, en inglés) tipo de entidad respecto a otra entidad más general.

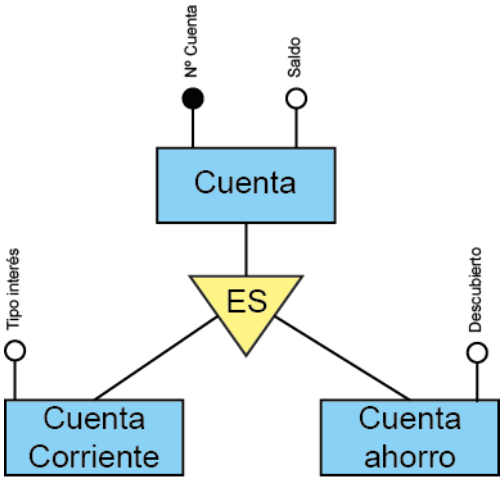
Se utilizan para unificar entidades agrupándolas en una entidad más general (generalización) o bien para dividir una entidad en entidades más específicas (especificación): aunque hoy en día a todas ellas se las suele llamar generalización e incluso (quizá incluso más adecuadamente) relaciones de herencia.

Se habla de superentidad refiriéndonos a la entidad general sobre las que derivan las otras (que se llaman subentidades). En la superentidad se indican los atributos comunes a todas las subentidades, se sobreentiende que las subentidades también tienen esos atributos, pero no se indican de nuevo esos atributos en el diagrama.

ESPECIALIZACIÓN

La especialización consiste en el proceso de designación de un subgrupo dentro de un determinado conjunto de entidades. Estos subgrupos serán designados atendiendo a los atributos específicos de cada subgrupo.

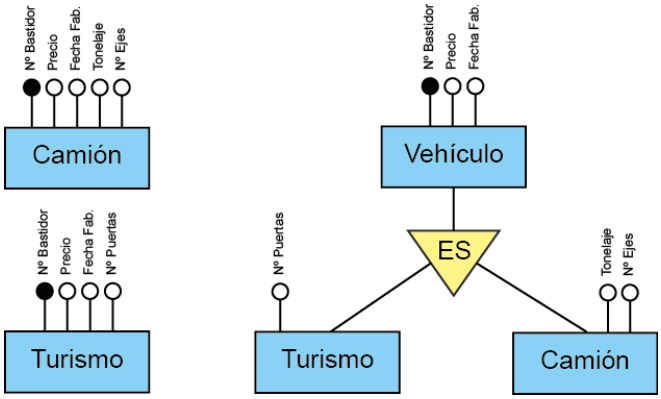
Un ejemplo de generalización sería una cuenta de un banco en la cual tenemos la entidad "Cuenta" que tiene una serie de atributos específicos, pero estas cuentas se pueden separar en dos grupos por un lado en una cuenta de ahorro y por otro en cuenta corriente, teniendo cada uno de estas subentidades una serie de atributos específicos que no se comparten entre ellas.



GENERALIZACIÓN

La generalización es una relación que permite expresar similitud entre un conjunto de entidades de nivel más bajo, sintetizando un conjunto de entidades de un nivel más alto a partir de atributos comunes. La generalización se obtiene mediante diseño ascendente; esto permite resaltar similitudes y economizar la representación ya que no se repiten atributos.

Por ejemplo un caso de generalización sería el caso de una base de datos en el que tenemos un catálogo de vehículos en el que podemos diferenciar los diferentes tipos de vehículos, podemos crear una clase superior que simplifique el número de atributos que tenemos en la base.



Las diferencias fundamentales entre la especialización y la generalización son principalmente dos:

• El punto de partida: la especialización se obtiene mediante un diseño descendente mientras que la generalización se obtiene mediante un diseño ascendente.

• El objetivo global: el objetivo de la especialización es diferenciar entidades, el de la generalización es sintetizarlas.

Normalmente cuando tenemos una especialización, las subentidades comparten clave con la superentidad (además de los atributos comunes); esto es muy matizable y de hecho hoy en día ningún diseñador intenta distinguir entre si tenemos una especialización o una generalización, porque al final ambas implican el mismo esquema interno en la base de datos.

En general se suelen indicar las cardinalidades en las relaciones ISA, pero se suele sobreentender (cuando no se indican explícitamente) que hay un (0,1) encima de cada subentidad (que significa que cada ejemplar de la subentidad solo puede relacionarse como mucho con uno de la subentidad e incluso con ninguno; un empleado de personal podría ser o no ser un profesor).

Lo recomendable es utilizar relaciones ISA cuando ocurre cualquiera de estas situaciones:

- Las subentidades tienen atributos distintos.
- Las subentidades tienen relaciones distintas.

2.2.2 Herencia

Tanto en la generalización como en la especialización se produce herencia de atributos. Los conjuntos de entidades de nivel más bajo (subclases) heredan los atributos y la participación en las relaciones de los conjuntos de entidades de nivel más alto (superclase).

Jerarquía

En una jerarquía un conjunto de entidades dado puede estar implicado como un conjunto de entidades de nivel más bajo sólo en una única relación ES.

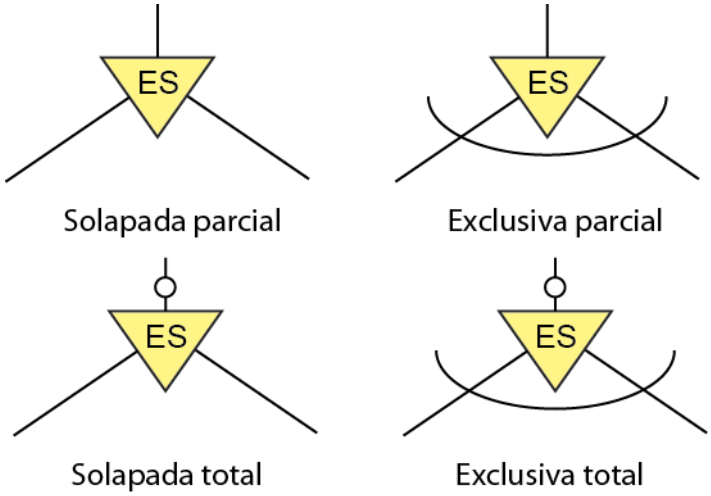
Herencia Múltiple

Si un conjunto de entidades es un conjunto de entidades de nivel más bajo en más de una relación ES, entonces el conjunto de entidades tiene herencia múltiple.

2.2.3 Tipos de relaciones ISA

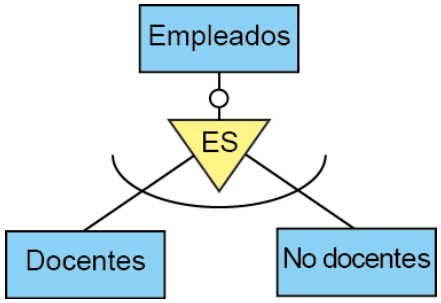
Los tipos de relaciones entre la superentidad y la subentidad que pueden aparecer en una relación ISA son fundamentalmente cuatro y se matiza en base a dos conceptos:

- **Obligatoriedad.** Indica si los ejemplares obligatoriamente se relacionan con ejemplares de las subentidades. Es decir cada entidad padre debe pertenecer al menos a una entidad hijo. Hay dos posibilidades:
    - Relaciones de jerarquía parcial. Indican que hay ejemplares de la superentidad que no se relacionan con ningún ejemplar de las subentidades.
    - Relaciones de jerarquía total. Indican que todos los ejemplares de la superentidad se relacionan con alguna subentidad. Se indican con un círculo encima del triángulo de la relación ISA.
- **Número de relaciones**. En este caso se mide con cuántas subentidades se relaciona la subentidad; es decir, si hay personal que pueda ser profesor y bedel a la vez o si sólo puede ser una cosa. Posibilidades:
    - Relaciones de jerarquía solapada. Indican que un ejemplar de la superentidad puede relacionarse con más de una subentidad. Ocurren cuando no hay dibujado un arco de exclusividad.
    - Relaciones de jerarquía exclusiva. Indican que un ejemplar de la superentidad sólo puede relacionarse con una subentidad. Ocurren cuando hay dibujado un arco de exclusividad.

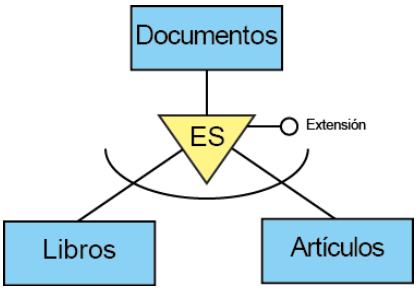


Ejemplos:

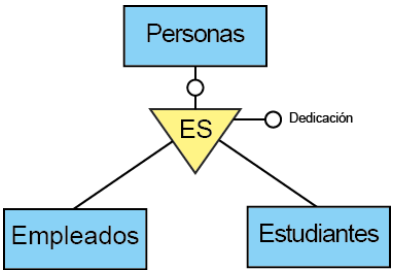
1. Un empleado es o bien docente o bien no docente (uno de los dos obligatoriamente). La pertenencia la determina el usuario, no un atributo.



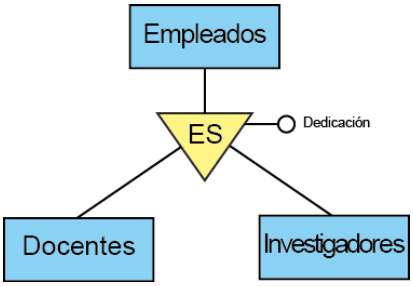
2. Según la extensión se determina si un documento es libro o artículo. Un artículo no puede ser un libro y viceversa (disjunto). Hay más tipos de documentos (no es total): revistas, etc.



3. El conjunto de entidades “Personas” es solapado: pueden ser empleados y estudiantes a la vez. Es total porque para la universidad sólo hay o empleados o estudiantes. La pertenencia viene definida por el atributo “Dedicación”.

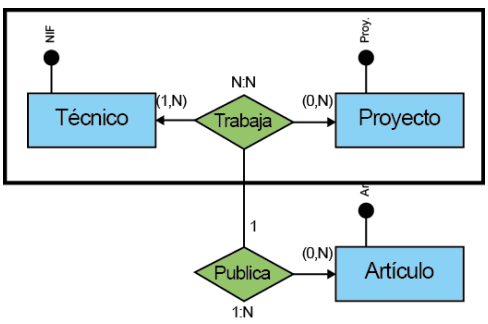


4. “Empleados” no es disjunto porque pueden ser ambas cosas a la vez (solapado). Como puede haber más tipos de empleados, es parcial y el atributo “Dedicación” determina la participación.



2.2.4 Agregaciones

La agregación surge de la limitación que existe en el modelado de E-R, al no permitir expresar las relaciones entre relaciones de un modelo E-R. Permite combinar varios tipos de entidad, relacionados mediante un tipo de relación, para formar un tipo de entidad agregada de nivel superior. Es útil cuando el tipo de entidad agregado debe relacionarse con otros tipos de entidad.

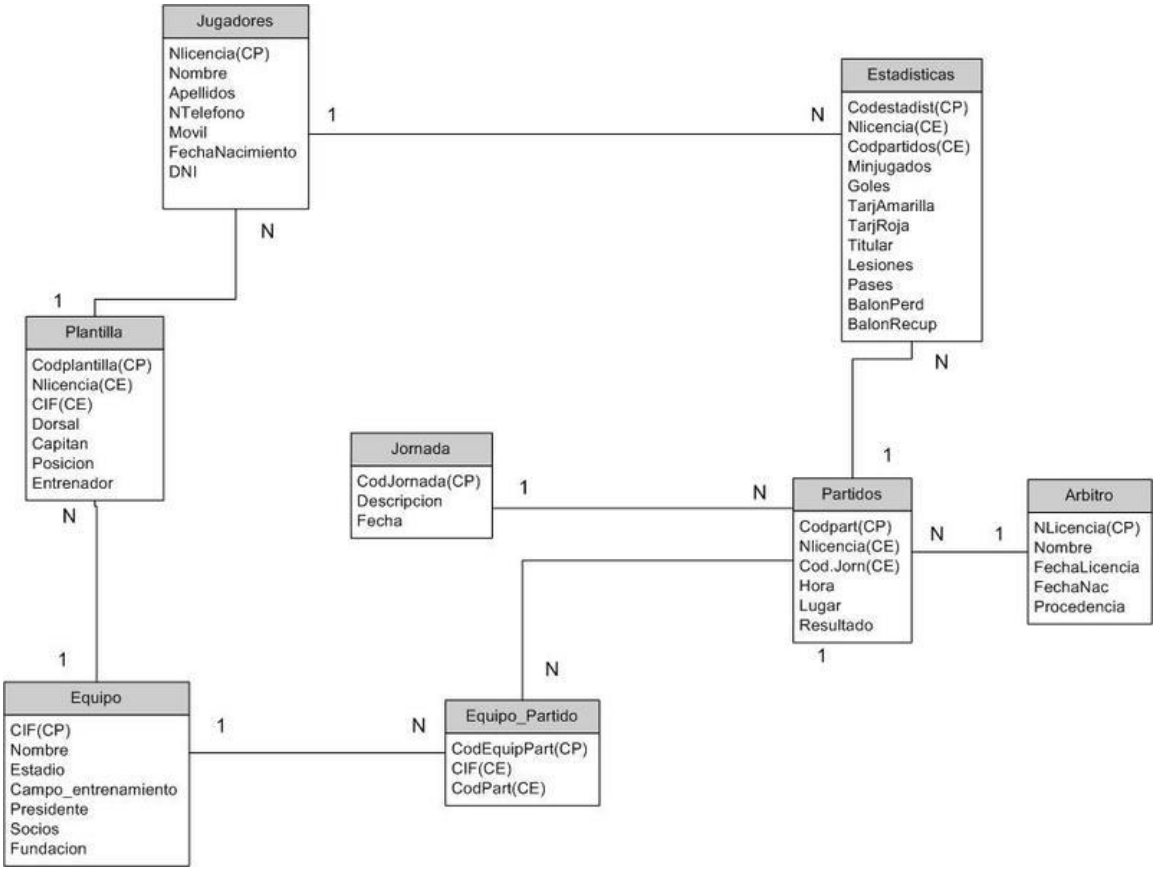


2.3 DIAGRAMA RELACIONAL

El modelo relacional, el cual se representa por medio de un diagrama relacional, constituye un sistema para poder organizar y representar la información que se pretende almacenar en una base de datos. Se trata de un modelo que, además de proporcionarnos los elementos básicos de modelado (las relaciones), incluye un conjunto de operadores para su manipulación, sin ambigüedad posible.

El carácter formal del modelo relacional hace relativamente sencilla su representación y gestión por medio de herramientas informáticas. La gran mayoría de los Sistemas de Gestión de Bases de Datos comerciales disponibles en el mercado utilizan este tipo de modelo para implementar las bases de datos que gestionan, proporcionando herramientas que permiten pasar directamente de una representación de diagrama relacional a base de datos.

En el modelo relacional se basa en el concepto matemático de relación. En este modelo, la información se representa en forma de "tablas" o relaciones, donde cada fila de la tabla se interpreta como una relación ordenada de valores (un conjunto de valores relacionados entre sí). El siguiente ejemplo presenta una relación que representa al conjunto de los departamentos de una determinada empresa, y que recoge información sobre los mismos.



Las ventajas que persigue la creación de un modelo relacional son:

- Independencia física de los datos: el modo de almacenamiento de los datos no debe influir en su manipulación lógica.
- Independencia lógica de los datos: los cambios que se realicen en los objetos de la base de datos no deben repercutir en los programas y usuarios que acceden a ella.

- Flexibilidad: para presentar a los usuarios los datos de la forma más adecuada.
- Uniformidad: en la presentación de la lógica de los datos, que son tablas, lo que facilita la manipulación de la base de datos por parte de los usuarios.
- Sencillez: este modelo es fácil de comprender y utilizar por el usuario.

La relación es el elemento básico del modelo relacional y se representa como una tabla, en la que se puede distinguir:

- El nombre de la tabla.
- El conjunto de columnas que representan las propiedades de la tabla y que se denominan atributos,
- El conjunto de filas, llamadas tuplas que contienen los valores que toman cada uno de los atributos para cada elemento de la relación.

Una relación tiene una serie de elementos característicos que la distinguen de una tabla:

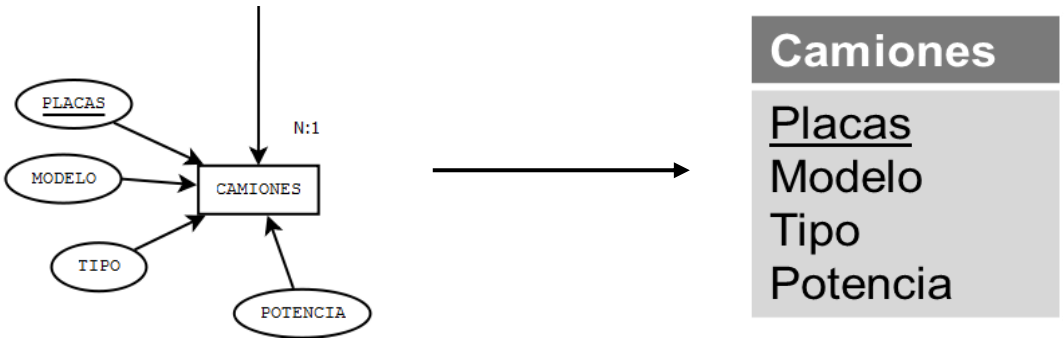
- No admiten filas duplicadas.
- Las filas y las columnas no están ordenadas.
- La tabla es plana. En el cruce de una fila y una columna solo puede haber un valor.

2.3.1 Reducción de E/R a Relacional

A la hora de reducir un diagrama de entidad relación a un diagrama relacional hay que tener presente cada uno de los tipos de elementos que podemos encontrar en el diagrama E/R. Así tenemos que cada elemento se corresponde con una tabla o conjunto de tablas.

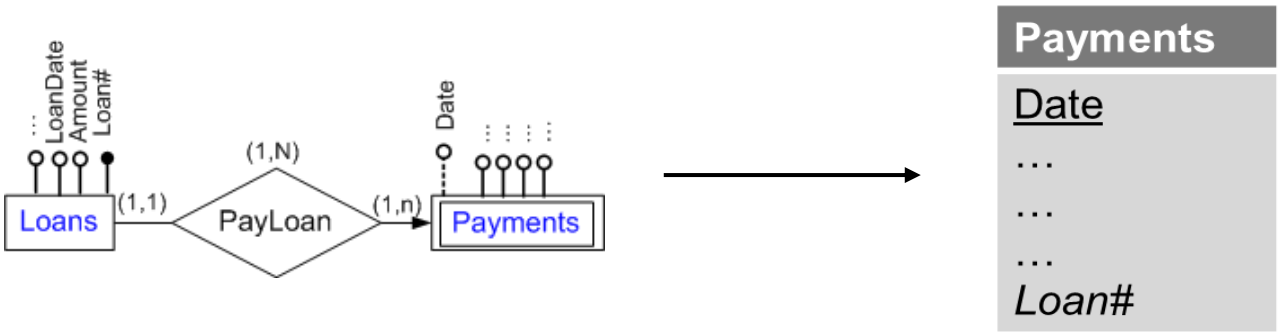
ENTIDADES FUERTES

Las entidades fuertes se reducen directamente a una tabla que contiene los mismos atributos que tiene en el diagrama E/R. La tabla resultado tendrá tantas columnas como atributos. En el caso de que se trate de un atributo compuesto, estos se separan creando un atributo por cada componente del atributo.



ENTIDADES DEBILES

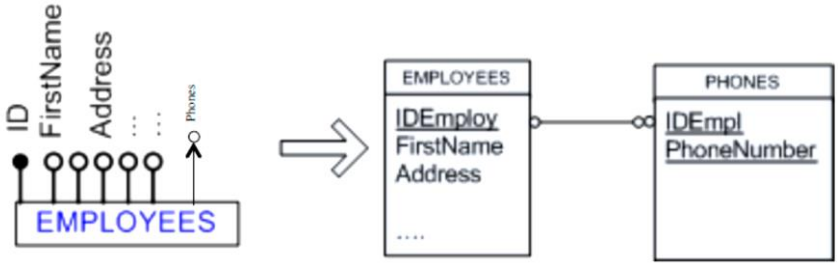
Se convierten en una tabla en la que cada atributo es un campo y se añade un campo en el que se incluye una referencia al identificador de la entidad fuerte.



ATRIBUTOS MULTIVALORADOS

Para un atributo de este tipo siempre se crea una nueva tabla. La nueva tabla tendrá un atributo correspondiente a la clave primaria de la entidad y un atributo correspondiente al atributo multivalorado.

Cada nuevo valor será una fila nueva de la tabla.



RELACIONES

En función de la relación existente entre las entidades la transformación al diagrama relacional es diferente, ya que implicará la creación de nuevas tablas o simplemente servirá con las tablas que se tienen creadas para los atributos. En función de la relación tenemos:

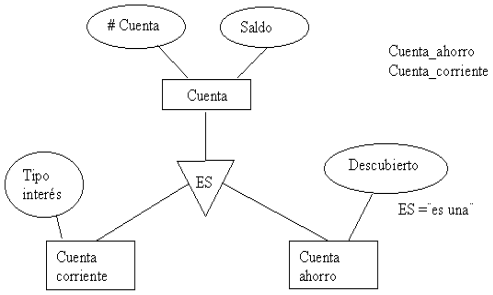
- Muchas a muchas: es necesario crear una nueva tabla con las columnas de las claves primarias de las dos entidades representantes de la relación y los atributos descriptivos de la relación.
- Muchas a una o una a muchas: cuando la participación es total se añade un atributo extra a la tabla correspondiente a la relación muchas que hace referencia a la clave primaria de la otra tabla. Si la participación es parcial puede ser mejor crear una tabla adicional.
- Una a una: Es igual que el caso anterior y se escoge cualquier tabla como si fuese la relación de muchas.

ESPECIALIZACIONES/GENERALIZACIONES

Existen dos métodos que se pueden aplicar para obtener la reducción de tablas:

- Metodo 1: Se crea una tabla para la entidad del nivel superior con todos sus atributos y a continuación se crea una tabla para cada entidad del nivel inferior con sus atributos en la que le incluyen la clave primaria del nivel superior.

- Método 2: Es utilizado cuando se trabaje con una generalización completa y disjunta. En este caso se crea una tabla para cada una de las entidades hijo con una columna para cada uno de sus atributos y otra columna para cada uno de los atributos del padre.



Cuenta
Cuenta Saldo

Cuenta corriente
Cuenta Tipo interés

Cuenta ahorro
Cuenta Descubierto

AGREGACIONES

A la hora de convertir una agregación se crea una tabla para el conjunto de relaciones que asocian el conjunto de entidades y la agregación. En esta tabla se debe incluir una columna para cada clave primaria de las entidades que participan en la agregación. Se debe incluir además las claves primarias de todas las entidades que estén relacionadas. Se debe incluir también cualquier conjunto de atributos descriptivos de las relaciones que intervienen en la agregación.

RESUMEN

Tipo de reducción	Número de tablas
Conjunto de entidades fuerte	1
Conjunto de entidades débiles	1
Conjunto de relaciones	El número depende de la redundancia existente y la combinación de tablas que se pueda realizar.
Atributos multivalorados	1
Generalización	Caso general: 1 tabla para el padre y otra para cada hijo. Generalización disjunta y completa: si interesa, 1 tabla para cada hijo.
Agregación	1 tabla para el conjunto de relaciones de la agregación cuya clave primaria será la del conjunto de entidades junto con la del conjunto de relaciones de la agregación (claves primarias de los dos conjuntos de entidades de la relación).

2.3.2 Restricciones

En todos los modelos de datos existen restricciones que a la hora de diseñar una base de datos se han de tener en cuenta. Los datos almacenados en la base de datos han de adaptarse a las estructuras impuestas por el modelo y deben cumplir una serie de reglas para garantizar que son correctas.

Los tipos de restricciones que aparecen en modelo relacional son:

INHERENTES AL MODELO

Indican las características propias de una relación que ha de cumplirse obligatoriamente, y que diferencian una relación de una tabla:

- No hay dos tuplas iguales.
- El orden de la tuplas y los atributos no es relevante.
- Cada atributo sólo puede tomar un único valor del dominio al que pertenece.
- Ningún atributo que forme parte de la clave primaria de una relación puede tomar un valor nulo.

SEMÁNTICAS

Como ya habíamos visto, una clave candidata es un conjunto de atributos que identifican unívoca y mínimamente cada tupla de la relación (es decir, cada fila de la misma). Sabemos que se deriva que siempre existe, al menos, una clave candidata y que si existe más de una clave candidata debemos distinguir entre:

- Clave Primaria (Primary Key): es la clave candidata que el usuario escoge para identificar las tuplas de la relación.
- Claves Alternativas (Alternative Key): las claves candidatas que no han sido escogidas como clave primaria.

Una clave ajena (Foreign Key) es una clave de una relación R2 a un conjunto no vacío de atributos cuyos valores han de coincidir con los valores de una clave candidata de una relación R1. Se define una clave ajena como aquel atributo de una tabla que es clave primaria en otra tabla con la cual está relacionada. Esta clave ajena sólo puede tomar valores que estén permitidos en la clave primaria, aunque no tienen porqué tener obligatoriamente el mismo nombre.

Las restricciones semánticas son definidas por el usuario y son las facilidades que el modelo ofrece a los diseñadores para que puedan reflejar en el esquema la semántica del mundo real. Los tipos de restricciones permitidas son:

- **Restricciones de clave primaria** (PRIMARY KEY): permiten declarar un atributo o un conjunto de atributos como clave primaria. Sus valores no se podrán repetir ni se admitirán los nulos.
- **Restricciones de unicidad** (UNIQUE): permiten declarar claves alternativas. Se debe tener en cuenta que todas estas claves alternativas tienen que ser distintas.
- **Restricciones de obligatoriedad** (NOT NULL): permiten determinar aquellos atributos que deben tomar obligatoriamente algún valor.
- **Restricciones de clave ajena** (FOREIGN KEY): estas claves ajenas se utilizan para enlazar tablas dentro de una base de datos siempre manteniendo la integridad referencial.

A la hora de definir las claves ajenas, hay que determinar las consecuencias que se producen al borrar o actualizar la relación referenciada. Estas consecuencias indican la acción que ocurre cuando se modifica la entidad padre que se realiza sobre la entidad hijo.

- **Borrado y/o modificación en cascada** (CASCADE): cuando se borra al padre se borran todos los hijos (siempre se habla del borrado de tuplas).
- **Borrado y/o modificación restringido** (RESTRICT): no se puede borrar al padre en caso de que tenga algún hijo.
- **Borrado y/o modificación con puesta a valores nulos** (SET NULLS): siempre que la clave ajena lo permita, al borrar al padre se pone la clave ajena a valor nulo.
- **Borrado y/o modificación con puesta a valor por defecto** (SET DEFAULT): al borrar al padre la clave ajena del hijo toma un valor por defecto que ya ha sido determinado con anterioridad a crear la relación.

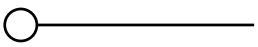
ADICIONALES

- **Restricción de verificación (**CHECK**):** con esta restricción se especifica una condición para los valores de un atributo cuando estos se introducen o actualizan en una tabla.
    - Se aplica a cada fila de la tabla.
    - Se debe verificar una condición
    - El resultado de la comprobación puede ser Verdadero, Falso o Desconocido. En caso de que el resultado sea Falso no se puede introducir ese valor.
- **Aserciones (**ASSERTION**):** especifican condiciones de valores de atributos en varias tablas. Se debe poner un nombre específico a cada una de las aserciones dado que no se sabe a qué tablas se aplican exactamente.
- **Disparadores (**TRIGGERS**):** es un código que se ejecuta automáticamente en respuesta a un determinado evento de la base.
    - Se utilizan para mantener la integridad de la información de una base.
    - También se utilizan para la actualización de atributos dependientes.
    - Un ejemplo serían las condiciones aplicadas en la inserción en tablas: al insertar un empleado nuevo en la tabla “Trabajadores” (más general) se quiere que lo inserte también en la de “Vendedores” (más específica). Se pueden considerar reglas ‘ECA’ (evento, condición, acción), según las cuales, dado un evento y cumpliendo una condición, se realiza una determinada acción.

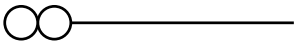
2.3.3 Notación en el diagrama relacional

- El nombre de la tabla se pondrá en mayúsculas.
- El nombre de la clave primaria debe estar subrayado.
- Cualquier valor asignado a una clave ajena debe pertenecer a los valores permitidos por la clave primaria a la que está asociada.
- Aquellos atributos cuyo valor NOT NULL venga impuesto por la reducción del diagrama Entidad/Relación a Relacional o como una restricción indicada en el enunciado del problema, se marcarán con un asterisco en el interior de un círculo.

- El lado en el que se encuentra la clave primaria se determinará con un círculo.

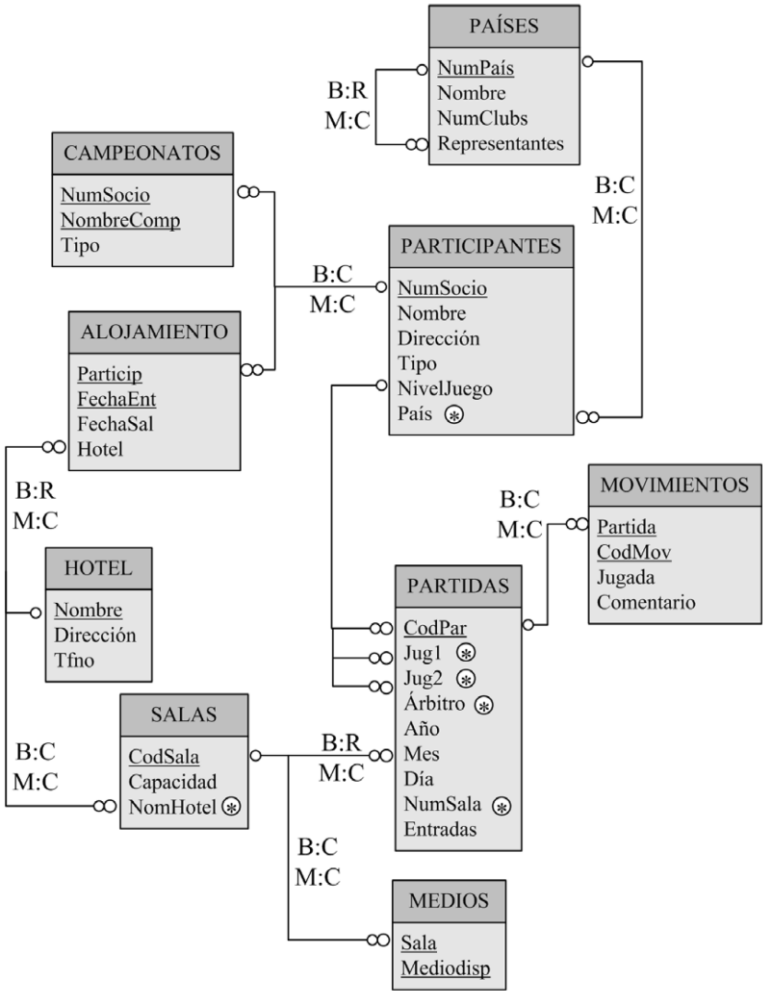


- El lado en el que se encuentra la clave ajena se denotará con un doble círculo.



- El borrado en cascada se denotará como B:C.
- El borrado restringido se denotará como B:R.
- El borrado con puesta a NULL se denotará como B:N.
- El borrado con puesta a valores por defecto se denotará como B:D.
- La modificación en cascada se denotará como M:C.
- La modificación restringida se denotará como M:R.
- La modificación con puesta a NULL se denotará como M:N.
- La modificación con puesta a valores por defecto se denotará como M:D.

Ejemplo



2.4 NORMALIZACIÓN

A la hora de crear bases de datos el diseño de las mismas es uno de los puntos más importantes ya que si la tabla no está bien diseñada, las consultas de los datos pueden devolver resultados enormemente complejos.

Uno de los puntos más importantes para el correcto diseño es normalizar el diseño de la base de datos, que permite eliminar redundancias e inconsistencias de dependencia en el diseño de las tablas. Para ello se define la normalización como:

Proceso mediante el cual se transforman datos complejos un conjunto de estructuras de datos más pequeñas.

- Hace las cosas fáciles de entender
- Hay menos repetición de datos, lo que implica un menor uso de espacio en disco
- Ayuda a prevenir errores lógicos en la manipulación de datos
- Facilita agregar nuevas columnas sin romper el esquema actual ni las relaciones.

A la hora de normalizar una base existen diferentes niveles de normalización y cada uno de ellos nos acerca más a hacer una base de datos verdaderamente relacional. Todos los niveles existentes son:

- Primera Forma Normal
- Segunda Forma Normal
- Tercera Forma Normal
- Forma Normal Boyce-Codd
- Cuarta Forma Normal
- Quinta Forma Normal o Forma Normal de Proyección-Unión
- Forma Normal de Proyección-Unión Fuerte
- Forma Normal de Proyección-Unión Extra Fuerte
- Forma Normal de Clave de Dominio.

Algunos de estos niveles implican una serie de reglas complejas que no se aplican a todas las bases sin embargo los tres primeros niveles son comunes a todos los diseños de tablas.

ID_ORDEN	FECHA	ID_CLIENTE	NOM_CLIENTE	ESTADO	NUM_ITEM	DESC_ITEM	CANT	PRECIO
2301	2/23/03	101	MARTI	CA	3786	RED	3	35
2301	2/23/03	101	MARTI	CA	4011	RAQUETA	6	65
2301	2/23/03	101	MARTI	CA	9132	PAQ-3	8	4.75
2302	2/25/03	107	HERMAN	WI	5794	PAQ-6	4	5.0
2303	2/27/03	110	WE-SPORTS	MI	4011	RAQUETA	2	65
2303	2/27/03	110	WE-SPORTS	MI	3141	FUNDA	2	10

Partamos de este ejemplo en el que estamos partiendo de una única tabla de base de datos en las que tenemos un nivel de normalización de cero en la cual no se han aplicado todavía las reglas de

normalización. En esta tabla evidentemente hay inconsistencias que implican que no se puede crear un sistema relacional de la base que sea eficiente.

2.4.1 Primera forma normal

La regla de la Primera Forma Normal establece que las columnas repetidas deben eliminarse y colocarse en tablas separadas. Una base de datos está en primera forma normal si cumple las siguientes condiciones:

- Sus tuplas (filas) contienen valores atómicos, es decir, no contienen valores que a su vez sean conjuntos.
- Incluye la eliminación de todos los grupos repetidos
- Todos los atributos (columnas) deben tener todos sus valores, o lo que es lo mismo, no debe haber celdas en blanco.
- Identifican cada grupo de datos relacionados con una clave primaria.

- ORDENES

ID_ORDEN	FECHA	ID_CLIENTE	NOM_CLIENTE	ESTADO
2301	2/23/03	101	MARTI	CA
2302	2/25/03	107	HERMAN	WI
2303	2/27/03	110	WE-SPORTS	MI

- ARTICULOS_ORDENES

ID_ORDEN	NUM_ITEM	DESC_ITEM	CANT	PRECIO
2301	3786	RED	3	35
2301	4011	RAQUETA	6	65
2301	9132	PAQ-3	8	4.75
2302	5794	PAQ-6	4	5.0
2303	4011	RAQUETA	2	65
2303	3141	FUNDA	2	10

Hemos divido la tabla en dos tablas, por un lado las ordenes y por otros los artículos de cada orden. De esta forma hemos eliminado todos los valores repetidos que teníamos en la primera tabla. Si nos fijamos en la primera tabla no podíamos establecer claramente una clave primaria puesto que los valores están repetidos. Al separar los datos en dos tablas en la tabla la primera podemos establecer un campo por cada cliente de tal manera que en esta tabla ya no existen valores repetidos. Al crear la segunda tabla ya no es necesario que el ID_ORDEN sea la clave primaria, sino que será una clave foránea, dejando como clave primaria otro campo y poder relacionarlo con la primera tabla.

En cuanto al resto de condiciones que debe cumplir una tabla en la primera forma normal, nos podemos dar cuenta que todas las celdas tienen valores atómicos, es decir, no existen varios valores en la misma celda y que todas las celdas tienen al menos un valor. Una vez que se ha cumplido la normalización de la tabla podemos establecer una clave primaria por cada una de las tablas que es el último requisito.

2.4.2 Segunda forma normal

La regla de la Segunda Forma Normal establece que todas las dependencias parciales se deben eliminar y separar dentro de sus propias tablas. Una dependencia parcial es un término que describe a aquellos datos que no dependen de la llave primaria de la tabla para identificarlos. Una base de datos cumple la segunda forma normal si se cumplen las siguientes condiciones:

- Es primera forma normal.
- Cualquier atributo (columna) no perteneciente a una clave (primaria o extranjera) tiene dependencia funcional total de la clave primaria, es decir, que a cada valor de dicho atributo solo le corresponde un valor de la clave primaria.
- Todas las dependencias parciales se deben eliminar y separar dentro de sus propias tablas.

- ARTICULOS_ORDENES

ID_ORDEN	NUM_ITEM	CANT
2301	3786	3
2301	4011	6
2301	9132	8
2302	5794	4
2303	4011	2
2303	3141	2

- ARTICULOS

NUM_ITEM	DESC_ITEM	PRECIO
3786	RED	35
4011	RAQUETA	65
9132	PAQ-3	4.75
5794	PAQ-6	5.0
3141	FUNDA	10

La tabla de ARTÍCULOS_ORDENES la hemos separado en dos porque no cumple la segunda forma normal. La razón de ello es que tenemos una tabla en la que estamos mezclando propiedades de datos, por un lado tenemos las características de los artículos que se están pidiendo y por otro lado tenemos las propiedades que son relativas a los propios artículos, eso nos lleva a que en esta tabla no se consiga establecer fácilmente la clave primaria, ya que existen dos columnas que nos están dando la misma información. La tabla ARTICULOS_ORDENES no se encuentra en 2FN ya que las columnas PRECIO y DESC_ITEM son dependientes de NUM_ITEM, pero no son dependientes de ID_ORDEN

En esta situación si tenemos que añadir un nuevo artículo que no se ha pedido nunca nos encontraríamos con la situación en la que no podríamos establecer todos los valores para ese nuevo registro. Si además tenemos que añadir nuevos campos que sean propiedad de los artículos habría que actualizar todos los pedidos.

Para ello en nuestro ejemplo hemos decidido extraer de la tabla de ARTÍCULOS_ORDENES las propiedades relativas a los artículos y crear una nueva tabla en la que se introducen todos estos valores. Una vez que tenemos separadas las dos tablas hay que crear una nueva relación entre ambas tablas. Para ello en

la nueva tabla hay que crear una clave primaria y en la tabla original hay que mantener una columna que sea clave foránea de la nueva tabla y que mantiene la relación.

Ahora mismo en la base tenemos ya tres tablas en lugar de una sola que era de la que partimos inicialmente, ya que la tabla de órdenes sigue existiendo.

2.4.3 Tercera forma normal

Una tabla está normalizada en esta forma si todas las columnas que no son llave son funcionalmente dependientes por completo de la llave primaria y no hay dependencias transitivas. Una relación es tercera forma normal si:

- Es segunda forma normal.
- Los atributos (columna) no pertenecientes a una clave (primaria o extranjera) son mutuamente independientes funcionalmente.

- ORDENES

ID_ORDEN	FECHA	ID_CLIENTE
2301	2/23/03	101
2302	2/25/03	107
2303	2/27/03	110

- CLIENTES

ID_CLIENTE	NOM_CLIENTE	ESTADO
101	MARTI	CA
107	HERMAN	WI
110	WE-SPORTS	MI

La tercera forma normal nos dice que tenemos que eliminar cualquier columna no llave que sea dependiente de otra columna no llave. Los pasos a seguir son en primer lugar determinar las columnas que son dependientes de otra columna no llave. A continuación se eliminan esas columnas de la tabla base y se crea una segunda tabla con esas columnas y con la columna no llave de la cual son dependientes.

Al observar las tablas que hemos creado, nos damos cuenta que tanto la tabla ARTICULOS, como la tabla ARTICULOS_ORDENES se encuentran en 3FN. Sin embargo la tabla ORDENES no lo está, ya que NOM_CLIENTE y ESTADO son dependientes de ID_CLIENTE, y esta columna no es la llave primaria. Para normalizar esta tabla, moveremos las columnas no llave y la columna llave de la cual dependen dentro de una nueva tabla CLIENTES.

2.4.4 Cuarta forma normal

La cuarta forma normal se centra en las relaciones existentes en la tabla que son de muchos a muchos. Se asegura de que las dependencias multivaluadas independientes estén correctas y eficientemente representadas en un diseño de base de datos.

Por tanto una base de datos está en cuarta forma normal si está en tercera forma normal y no posee dependencias multivaluadas no triviales. Para resolver este tipo de dependencias se crea una tabla intermedia en la relación como ya vimos.

2.4.5 Quinta forma normal

Existe otro nivel de normalización que se aplica a veces, aunque resulta un poco complejo y en la mayoría de los casos no es necesario para obtener la mejor funcionalidad de nuestra estructura de datos o aplicación.

- La tabla original debe ser reconstruida desde las tablas resultantes en las cuales ha sido troceada.

Los beneficios de aplicar esta regla aseguran que no has creado ninguna columna extraña en tus tablas y que la estructura de las tablas que has creado sea del tamaño justo que tiene que ser. Es una buena práctica aplicar esta regla, pero a no ser que se esté tratando con una extensa estructura de datos probablemente no es necesario.

2.4.6 Ventajas y desventajas de la normalización

Ventajas

- Facilita la compresión global del modelo de base de datos porque la información está agrupada de forma lógica.
- Ahorra espacio en disco porque se controla la redundancia de los datos.
- Es más sencillo mantener la integridad de las referencias porque no hay datos redundantes.
- Se obtienen datas más simples por lo que los índices se crean de forma más sencillas y el acceso a los datos también es más rápido.

Desventajas

- Las consultas de búsqueda son más costosas debido a que es necesario consultar un mayor número de tablas para obtener los datos.