2020

GRUPO STUDIUM ciclos

Tema 2: Temario

EL MODELO RELACIONAL

STUDIUM

www.grupostudium.com
informacion@grupostudium.com
954 539 952

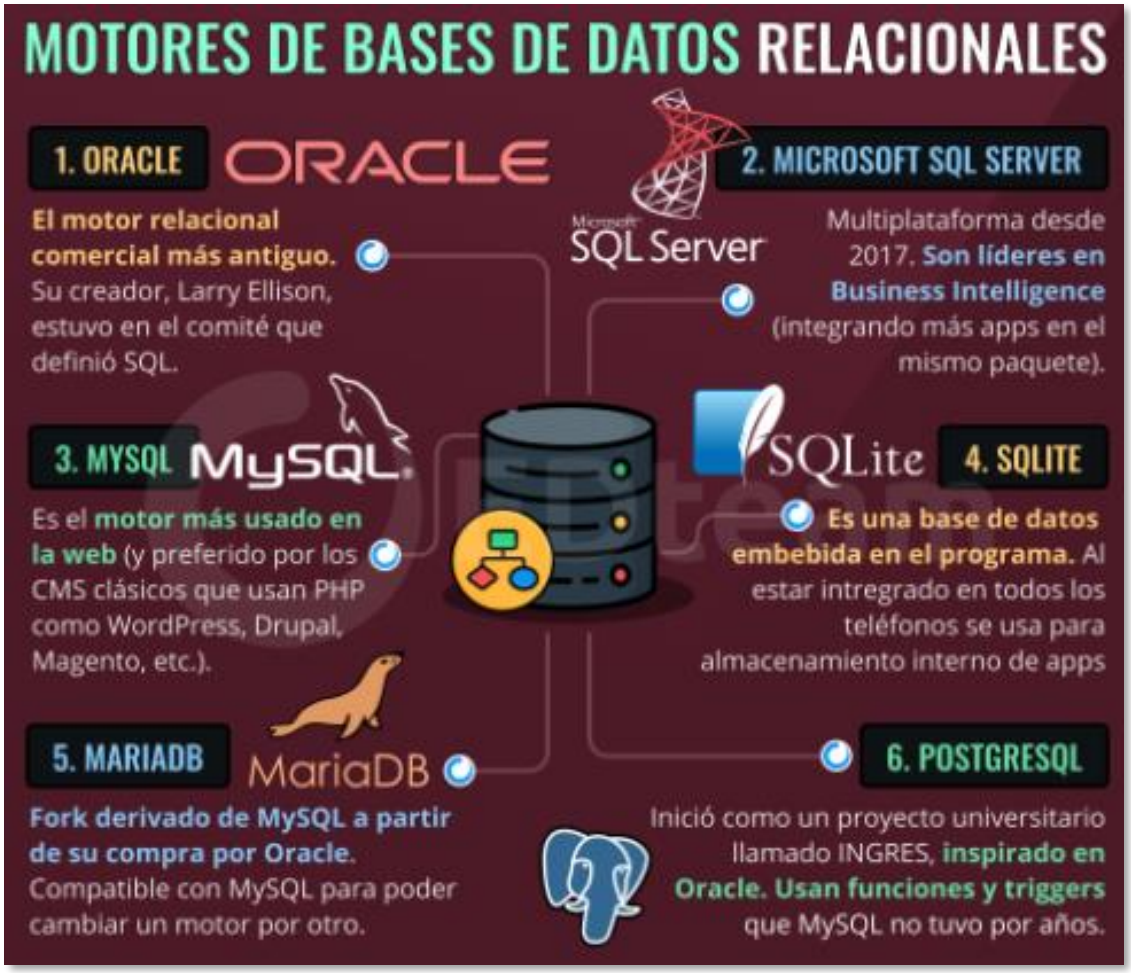
2.1 Introducción

Veamos una serie de definiciones que nos servirán para una mejor comprensión del resto del temario:

- **Base de datos o BD (DB o Data Base** en inglés**):** Colección de datos relacionados entre sí, estructurados (mediante campos) y organizados (ordenados).
- **Motor de la base de datos:** Aplicación que gestiona una base de datos a nivel básico.
- **Sistema Gestor de Bases de datos o SGBD (DBMS. o Data Base Management System):** Un conjunto de programas que acceden a los datos de una base de datos y permiten gestionarlos de una forma mucho más cómoda.
- **Incoherencia o inconsistencia:** En una Base de Datos, existe incoherencia o inconsistencia en sus datos cuando diversas copias del mismo dato no concuerdan entre sí.
- **El subsistema de integridad:** En un SGBD, este mecanismo se encarga de detectar y corregir en la medida de lo posible operaciones incorrectas que pueden crear inconsistencias. Vigila la integridad de la Base de Datos.
- **Reglas de validación de un campo:** Expresiones que limitan los valores que pueden almacenarse en un campo. Por ejemplo, en Access, un campo numérico que únicamente puede tomar valores de 1 a 99, podría tener la regla de validación siguiente: ">0 Y <100".
- **Máscara de entrada de un campo:** Formato general o patrón físico que deben respetar los datos que se inserten en ese campo. Por ejemplo, en Access, un código postal podría ponerse como un campo de texto de 5 caracteres con la siguiente máscara de entrada "00000" (5 cifras numéricas obligadas).

El **objetivo primordial** del SGBD es proporcionar eficiencia y seguridad a la hora de extraer o almacenar información en el motor de la Base de datos. Permiten gestionar grandes bloques de información, incluyendo las estructuras de los datos para su almacenamiento y los mecanismos para su gestión.

www.grupostudium.com
informacion@grupostudium.com
954 539 952



El **SGBD** es una aplicación que permite a los usuarios definir, crear y mantener la Base de Datos, y proporciona un acceso controlado a la misma. Debe prestar los siguientes servicios:

- **Creación y definición de la Base de Datos**. El SGBD permite la especificación de la estructura, el tipo de los datos, las restricciones y las relaciones entre ellos mediante "lenguajes de definición de datos". Toda esta información se almacena en el diccionario de datos.
- **Manipulación de los datos**. El SGBD permite realizar consultas sobre los datos almacenados, inserciones de nuevos datos, y actualizaciones sobre los datos ya existentes, utilizando "lenguajes de manipulación de datos".
- **Acceso controlado a los datos de la Base de Datos**. El SGBD permite mantener la seguridad para controlar el acceso no autorizado a los datos por usuarios sin permisos establecidos.
- **Mantenimiento de la integridad y consistencia**. El SGBD permite mantener la coherencia de los datos a través de mecanismos correctores que controlan los cambios no autorizados en los datos.
- **Acceso compartido a la Base de Datos**. El SGBD permite la interacción entre usuarios concurrentes.

- **Mecanismos de copias de seguridad y recuperación de los datos**. El SGBD permite, tras un fallo del sistema que ha provocado errores en los datos, recuperar los datos desde algún sistema de almacenamiento externo donde se realizó una copia de seguridad recientemente.

Todos los motores de bases de datos tienen sus SGBD asociados; en ocasiones, más de uno. Aquí vemos los principales:

Motor Base de Datos	SGBD habitual
MySQL Server	MySQL Workbench
SQL Server	SQL Server Management Studio
MongoDB	Compass

2.2 Componentes de los SGBD

A continuación, vamos a ver una serie de elementos que nos van a permitir trabajar con los SGBD.

Lenguajes de los SGBD

Todos los SGBD ofrecen lenguajes e interfaces apropiadas para cada tipo de usuario: administradores, diseñadores, programadores de aplicaciones que acceden a la BD y usuarios finales. Dichos lenguajes se clasifican en:

1. **Lenguaje de Definición de Datos** (LDD o DDL): Permiten especificar el esquema conceptual e interno de la Base de Datos, es decir, crea las entidades, describe sus atributos, etc... Es utilizado por diseñadores y administradores de la Base de Datos.
2. **Lenguaje de Manipulación de Datos** (LMD o DML): Permiten leer y actualizar los datos de la Base de Datos. Los objetos creados en la Base de Datos con un lenguaje de definición de datos se gestionan con un lenguaje de manipulación de datos. Los usuarios lo utilizan para realizar consultas, inserciones, eliminaciones y modificaciones sobre los datos de la Base de Datos. Las Base de Datos relacionales utilizan lenguajes como SQL (Structured Query Language) o QBE (Query By Example).
3. **Lenguaje de control de datos** (LCD o DCL): Permiten conceder o suprimir privilegios a los usuarios, es decir, realiza el control del acceso a los datos. Con este lenguaje se establecen las vistas de los usuarios, así a cada usuario se le permite manipular únicamente el conjunto de datos que le interesan, y se le deniega el acceso a los datos que no necesita.
4. **Lenguajes de control de transacciones**: Controlan los cambios realizados en los datos de la Base de Datos mediante instrucciones de manipulación de datos (DML). Permiten agrupar varias instrucciones DML como si fuesen una única instrucción lógica de manipulación, de forma que o se realiza el grupo de instrucciones completo o se deshacen los cambios de todo el grupo (esto ocurriría

en el caso de que simplemente una de las instrucciones DML del grupo no pudiera realizarse o generara algún error en su ejecución).

5. **Lenguajes de cuarta generación** (4GL) o **Herramientas de Desarrollo**: La mayoría de los SGBD comerciales los incluyen, y permiten al usuario crear aplicaciones de forma fácil y rápida para acceder y manipular los datos de la Base de Datos.

Por ejemplo, **SQL Forms de Oracle**, que permite crear formularios para interactuar con los datos; **SQL Reports de Oracle**, para generar informes de los datos contenidos en la Base de Datos; **PL/SQL de Oracle**, que es un lenguaje que permite crear procedimientos que interactúen con los datos de la Base de Datos.

El Diccionario de Datos

Es el lugar donde se guarda información acerca de todos los datos que forman la Base de Datos: su descripción y la de los objetos que la forman. En una Base de Datos relacional, el diccionario de datos proporciona información acerca de:

- La estructura lógica y física de la Base de Datos. Esquemas externo, conceptual e interno, y correspondencia entre los esquemas.
    - Las definiciones de todos los objetos de la Base de Datos: tablas, vistas, índices, disparadores, procedimientos, funciones, etc.
    - El espacio asignado y utilizado por los objetos.
- Los valores por defecto de las columnas de las tablas.
- Información acerca de las restricciones de integridad.
- Los privilegios y roles otorgados a los usuarios.
- Estadísticas de utilización, tales como la frecuencia de las transacciones y el número de accesos realizados a los objetos de la base de datos.
- Se puede tener un historial de los cambios realizados sobre la base de datos.

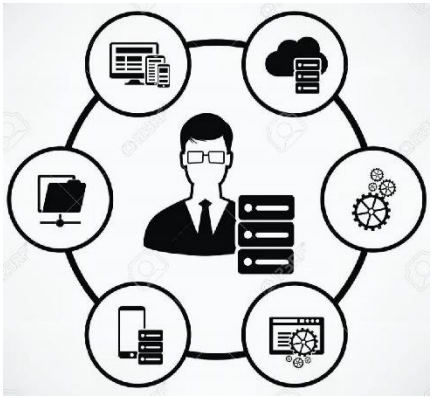
Seguridad e Integridad de los Datos

Un SGBD proporciona los siguientes mecanismos para garantizar la seguridad e integridad de los datos:

- Debe garantizar la **protección** de los datos contra accesos no autorizados, tanto intencionados como accidentales.
- Debe implantar **restricciones** de integridad que protegerán la Base de Datos contra daños accidentales, pues los valores de los datos que se quieren almacenar deberán satisfacer ciertos tipos de restricciones de consistencia y reglas de integridad, especificadas por el administrador de la Base de Datos. El SGBD puede determinar si se produce una violación de la restricción impuesta.

- Debe proporcionar herramientas y mecanismos para la **planificación** y **realización** de copias de seguridad, y su posible posterior restauración tras un fallo del sistema.
- Debe ser capaz de **recuperar** la Base de Datos llevándola a un estado consistente en caso de ocurrir algún suceso que la dañe.
- Debe asegurar el **acceso concurrente** y ofrecer mecanismos para conservar la consistencia de los datos en el caso de que varios usuarios actualicen la Base de Datos de forma concurrente.

El Administrador de la Base de datos



En los SGBD existen distintos tipos de usuarios, cada tipo con unos permisos o privilegios diferentes sobre los objetos que forman la Base de Datos.

El **DBA** o **Data Base Administrator** tiene una gran responsabilidad al tener el máximo nivel de privilegios. Hay que intentar que haya solo uno o muy pocos. Los DBA crearán los demás usuarios y les asignarán sus tipos y permisos de acceso.

Entre las tareas de un DBA están:

1. Instalar el SGBD en el sistema informático de la empresa.
2. Crear las BBDD que se vayan a gestionar.
3. Crear y mantener el esquema de la Base de Datos.
4. Crear y mantener las cuentas de usuario de la Base de Datos.
5. Arrancar y parar el SGBD, y cargar las Bases de Datos con las que se ha de trabajar.
6. Colaborar con el Administrador del SO en las tareas de ubicación, dimensionado y control de los archivos y espacios de disco ocupados por el SGBD.
7. Colaborar en las tareas de formación de los usuarios.
8. Establecer estándares de uso, políticas de acceso y protocolos de trabajo diario para los usuarios de la Base de Datos.
9. Suministrar la información necesaria sobre la Base de Datos a los equipos de análisis y programación de aplicaciones.
10. Efectuar tareas de explotación como:
    - 10.1 Vigilar el trabajo diario colaborando en la información y resolución de las dudas de los usuarios.
    - 10.2 Controlar en tiempo real los accesos, tasas de uso, cargas en los servidores, anomalías, etc.
    - 10.3 Llegado el caso, reorganizar la Base de Datos.
    - 10.4 Efectuar las copias de seguridad periódicas de la Base de Datos.

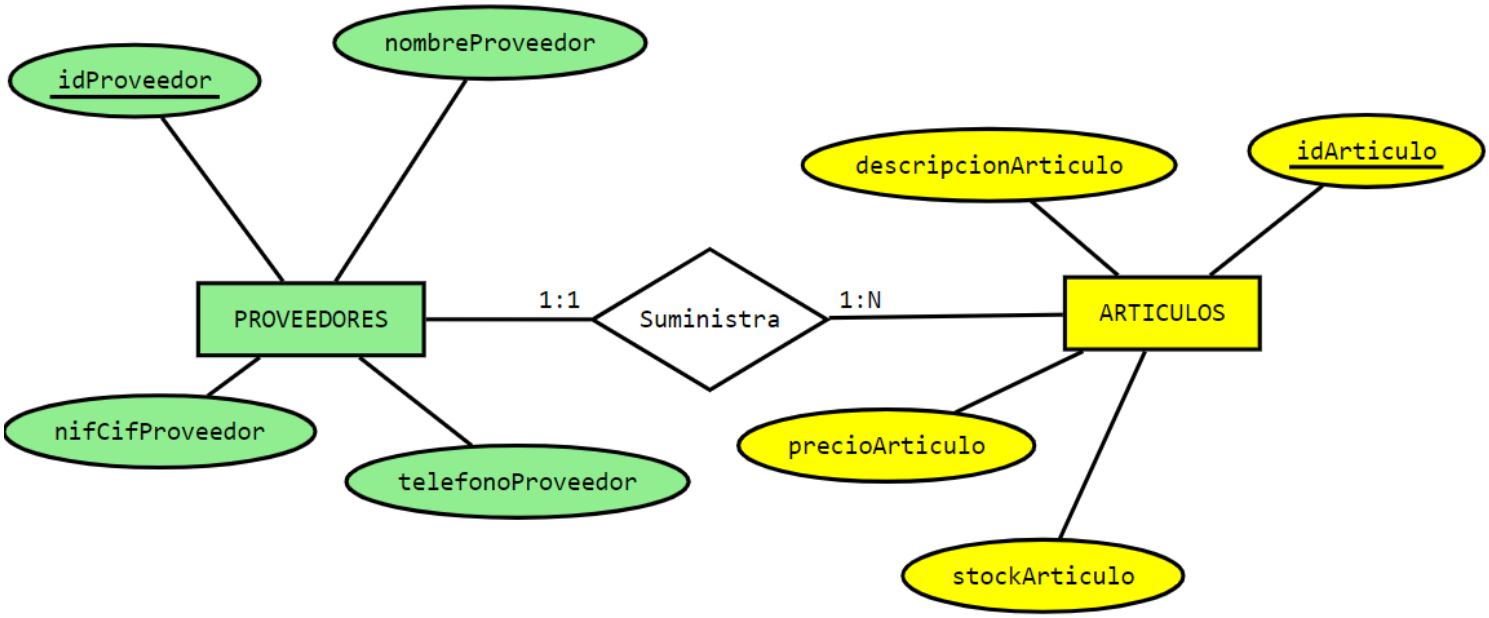
10.5. Restaurar la Base de Datos después de un incidente material a partir de las copias de seguridad.

10.6. Estudiar las auditorías del sistema para detectar anomalías, intentos de violación de la seguridad, etc.

10.7. Ajustar y optimizar la Base de Datos mediante el ajuste de sus parámetros, con ayuda de las herramientas de monitorización y de las estadísticas del sistema.

2.3 El Modelo de datos Entidad-Relación

El **Modelo Entidad-Relación** (o Modelo E-R o Modelo Entidad-Interrelación) fue propuesto por Peter Chen en 1976 para la representación conceptual de los problemas del mundo real. Este modelo de datos representa los datos utilizando grafos y símbolos gráficos, además de tablas para la representación de los datos y sus relaciones.



Conceptos básicos usados en el Modelo E-R

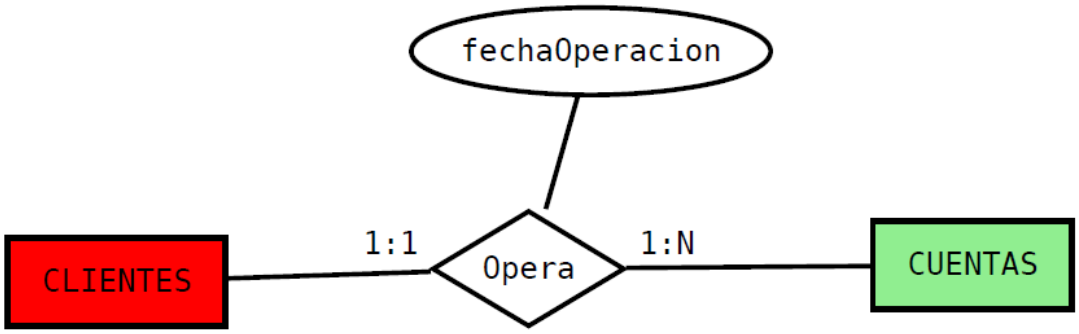
- **Entidad**: Es un objeto del mundo real que tiene interés para la empresa. Por ejemplo, la entidad ALUMNOS de un centro escolar o la entidad CLIENTES de una empresa. Se representan con rectángulos con el nombre en el interior.
- **Entidad Fuerte**: Es una entidad que no depende de otra entidad para su existencia. Por ejemplo, la entidad ALUMNOS es fuerte pues no depende de

otra para existir como entidad, mientras que la entidad NOTAS es una entidad débil pues necesita a la entidad ALUMNOS para existir.

- **Atributos o Campos**: Son las unidades de información que describen propiedades de las entidades. Por ejemplo, la entidad ALUMNOS posee los atributos: número de matrícula, nombre, dirección, población, código postal, provincia, y teléfono. Los atributos toman valores, por ejemplo, el atributo provincia podría ser SEVILLA, CÁDIZ, etc. Se representan mediante una elipse con el nombre en el interior.
- **Dominio**: Es el conjunto de valores permitidos para cada atributo. Por ejemplo, el dominio del atributo *nombreAlumno* puede ser el conjunto de cadenas de texto de una longitud determinada.
- **Identificador o Superclave**: Es el conjunto de atributos que identifican de forma única a cada entidad. Por ejemplo, la entidad EMPLEADOS, con los atributos: número de la seguridad social, dni, nombre, dirección, fecha de nacimiento y teléfono, podría tener como identificador sólo el dni (pues no habrá 2 empleados con el mismo dni), o sólo el número de la seguridad social, o el conjunto de 3 atributos nombre, fecha de nacimiento y teléfono (pues es difícil que hay 2 empleados en la misma empresa que tengan los mismos valores en esos 3 atributos).
- **Clave Candidata**: Es cada una de las superclaves formadas por el mínimo número de campos posibles. En el ejemplo anterior habría 2 claves candidatas de un único atributo: dni o número de la seguridad social.
- **Clave Primaria o Clave Principal (Primary Key)**: Es la clave candidata seleccionada por el diseñador de la Base de Datos para identificar a cada entidad. Una clave primaria no puede tener valores nulos (vacíos), ha de ser sencilla de crear y no ha de variar con el tiempo. El atributo o conjunto de atributos que forman parte de la clave primaria se representan subrayados.
- **Clave Ajena o Clave Foránea (Foreign Key):** Es el atributo o conjunto de atributos de una entidad que constituyen la clave primaria de otra entidad. Las claves foráneas representan las relaciones entre entidades. Por ejemplo, la entidad ARTICULOS con los atributos: código de artículo, descripción de artículo, precio de venta y stock en almacén, y la entidad VENTAS con los atributos: código de venta, fecha de venta, código de artículo y unidades vendidas; pues el atributo código de artículo es clave foránea en la entidad VENTAS, pues la relaciona con la entidad ARTICULOS, debido a que ese atributo es clave primaria de la entidad ARTICULOS.
- **Relación**: Es una asociación entre diferentes entidades. Se representan mediante un rombo con su nombre, un verbo, en su interior.
- **Conjunto de Relaciones**: Es un grupo de relaciones del mismo tipo. Por ejemplo, entre los conjuntos de entidades ARTICULOS y VENTAS puede haber

varias relaciones distintas, pues todas ellas pueden formar un conjunto de relaciones, que vinculan el conjunto de entidades ARTICULOS con el de VENTAS.

Una **relación** puede tener atributos descriptivos, por ejemplo, supongamos que la entidad CLIENTES está relacionada con la entidad CUENTAS a través de una relación OPERA; se necesitaría el atributo *fechaOperacion* en el conjunto de relaciones CLIENTES_CUENTAS, que especificaría la última fecha en la que el cliente tuvo acceso a su cuenta bancaria.



Una **metodología** es un conjunto de modelos y herramientas que nos permitan pasar de una etapa a la siguiente en el proceso de diseño de la base de datos. Estos pasos los podemos resumir en los siguientes:

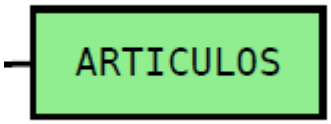
- Creación de una base de datos
- Estado previo y plan de trabajo
- Concepción de la base de datos y selección del equipo
- Diseño y carga de datos
- Producción

En resumidas cuentas, el Modelo E/R se basa, fundamentalmente en **Entidades**, **Atributos** y **Relaciones**.

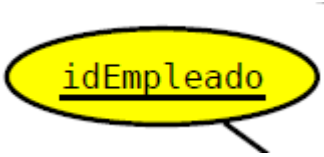
Diagramas de estructuras de datos en el modelo E-R

Los diagramas E-R representan la estructura lógica de una Base de Datos de manera gráfica. Los símbolos utilizados son:

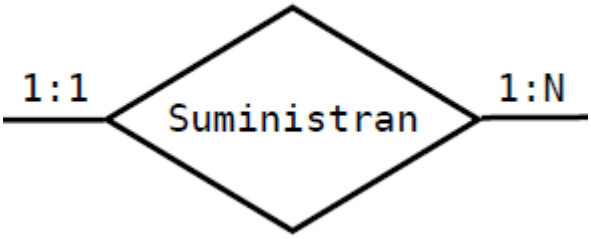
1. Rectángulos para representar entidades.



2. Elipses para los atributos.



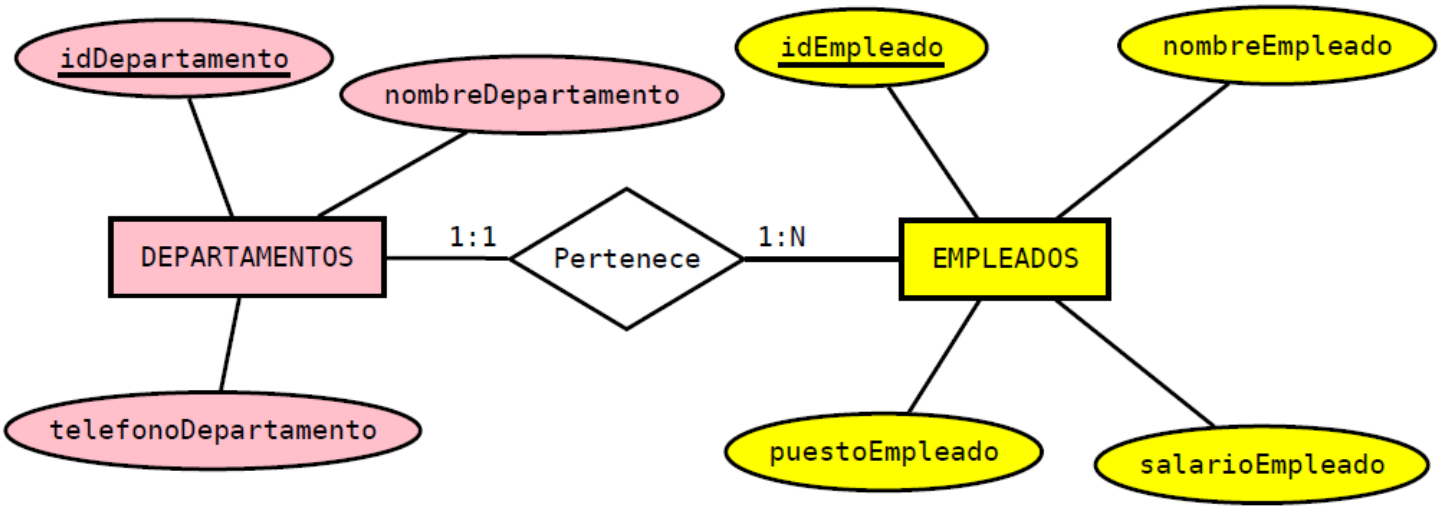
3. Rombos para las relaciones.



4. Cada atributo se unirá a la entidad o a la relación a la que pertenezca con líneas simples.

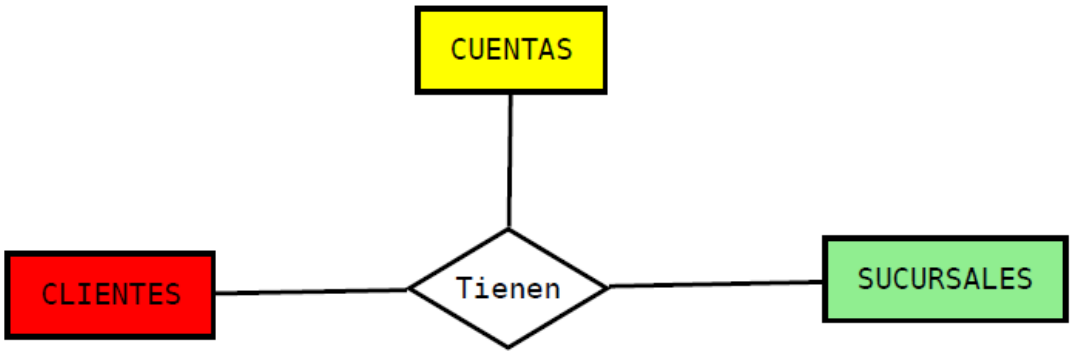
5. Las líneas podrán tener forma de flecha en una relación. Donde esté la punta de la flecha estará el MUCHOS (N), y donde no hay punta de flecha en la línea estará el UNO (1). La orientación de la flecha señala la cardinalidad de la relación.

6. Cada componente gráfico se etiqueta con el nombre que lo representa.



Grado y cardinalidad de las relaciones

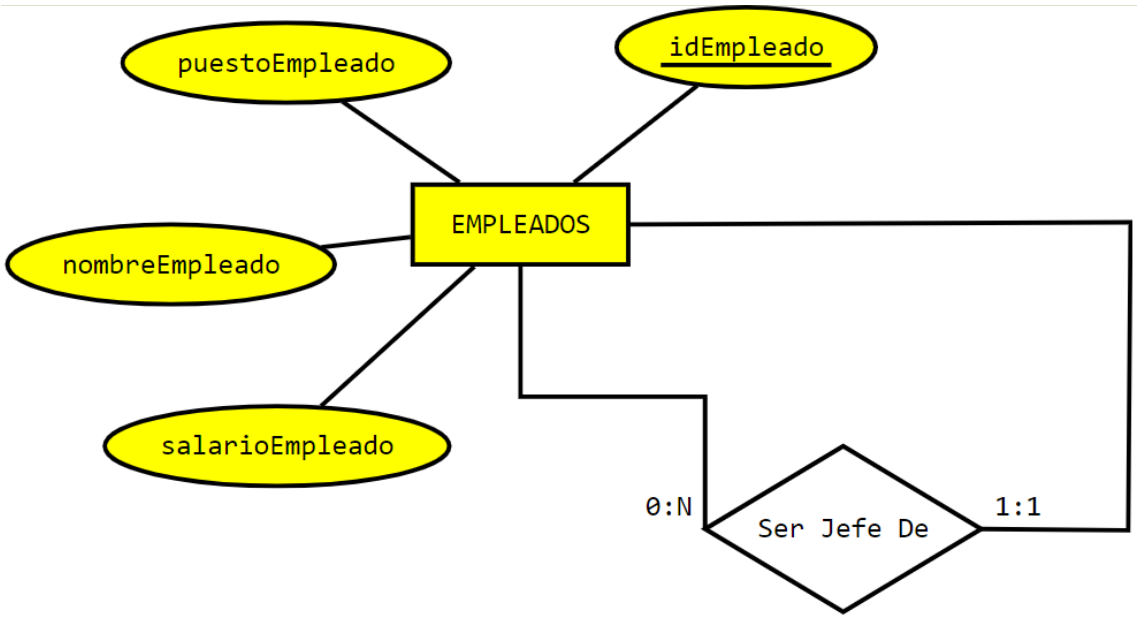
El **grado** de una relación es el número de entidades que participan en el conjunto de relaciones, es decir, el número de entidades que participan en una relación. Lo normal es que las relaciones sean binarias (relaciones de grado 2), es decir, que en las relaciones participen 2 entidades. No obstante, puede haber relaciones ternarias (de grado 3) o incluso de otro grado, aunque son poco comunes. Las relaciones en las que sólo participa una entidad se llaman anillo o de grado 1 o relaciones reflexivas.



Ejemplo de Relación Ternaria

Las relaciones entre 3 entidades se pueden descomponer, a veces, en 2 relaciones de 2 entidades.

Un ejemplo de relación de anillo sería el siguiente: la entidad EMPLEADOS puede tener una relación SER JEFE DE consigo misma, pues un empleado es jefe de muchos empleados y, a la vez, el jefe es un empleado. Otro ejemplo sería la relación SER DELEGADO DE los alumnos de un curso, pues el delegado es también alumno del curso.

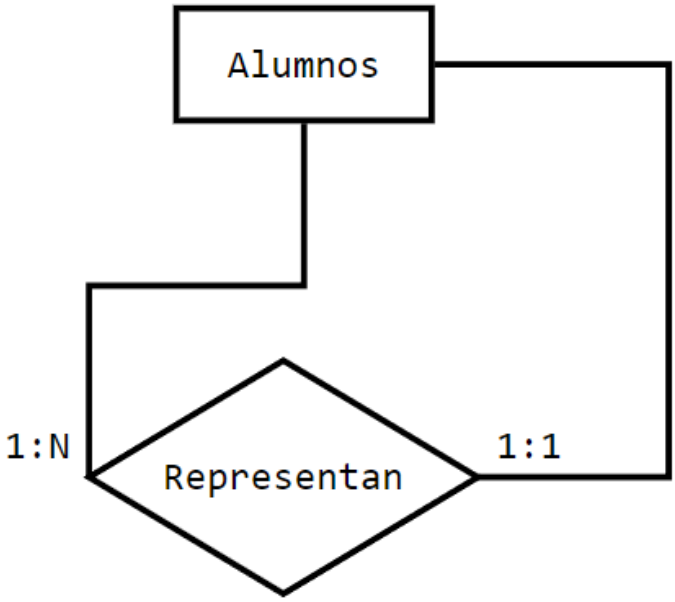


www.grupostudium.com
informacion@grupostudium.com
954 539 952

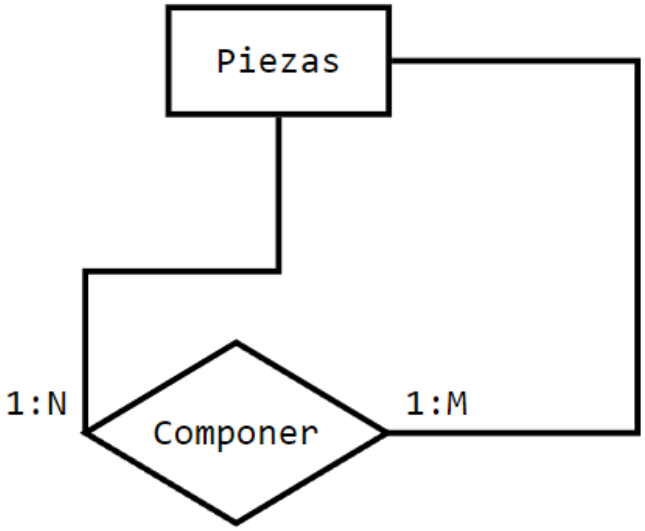
La relación SER JEFE DE asocia la entidad EMPLEADOS consigo misma. Es una relación de grado 1, es decir, **reflexiva**. Su cardinalidad es 1:N por lo siguiente:

- Un empleado tiene un jefe y sólo uno (1:1)
- Un empleado es jefe de cero o más empleados (0:N)

Otro ejemplo de relación **reflexiva** podría ser la que se establece entre la entidad Alumnos y el Representante o Delegado de los mismos, que no dejan de ser Alumnos igualmente. Se representaría de esta forma:



Un último ejemplo de relación reflexiva: Tenemos la entidad Piezas que está compuesta por otras Piezas.

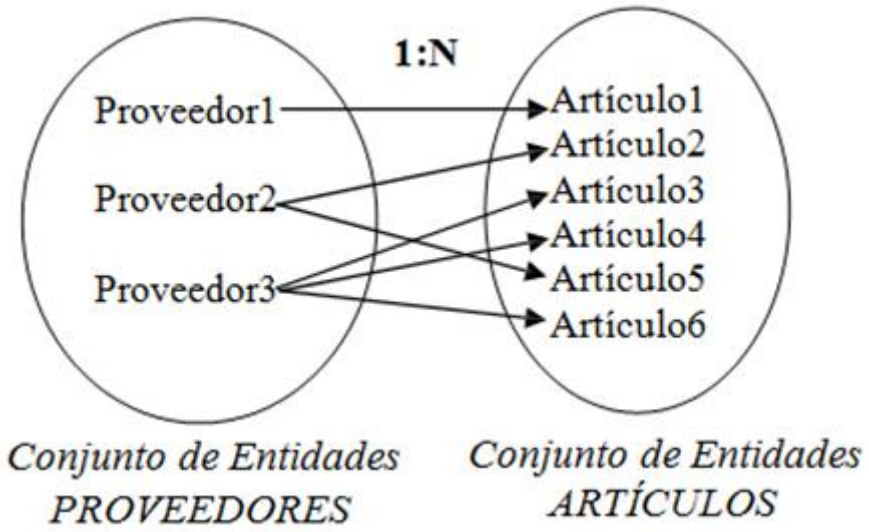


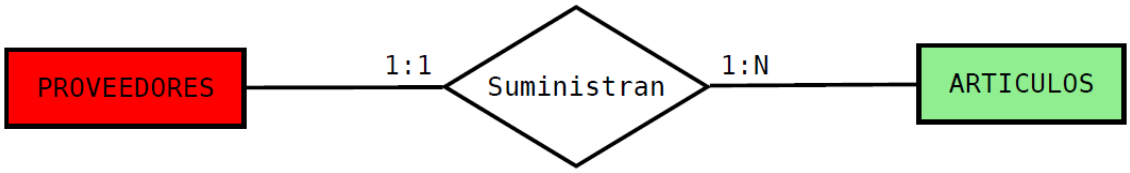
Las **cardinalidades de asignación** expresan el número de entidades a las que puede asociarse otra entidad mediante un conjunto relación. Las cardinalidades de asignación se describen únicamente para conjuntos binarios de relaciones.
Las cardinalidades de asignación son las siguientes:

1. **1:1 o uno a uno**: A cada elemento de la primera entidad le corresponde sólo uno de la segunda entidad, y a la inversa. Por ejemplo, un cliente de un hotel ocupa una habitación y cada habitación es ocupada por un cliente titular; o, por ejemplo, cada curso de alumnos tiene un único tutor, y ese tutor es únicamente tutor de ese curso.

Ejemplo de relación con cardinalidad 1:1, es decir, es una relación de uno a uno.

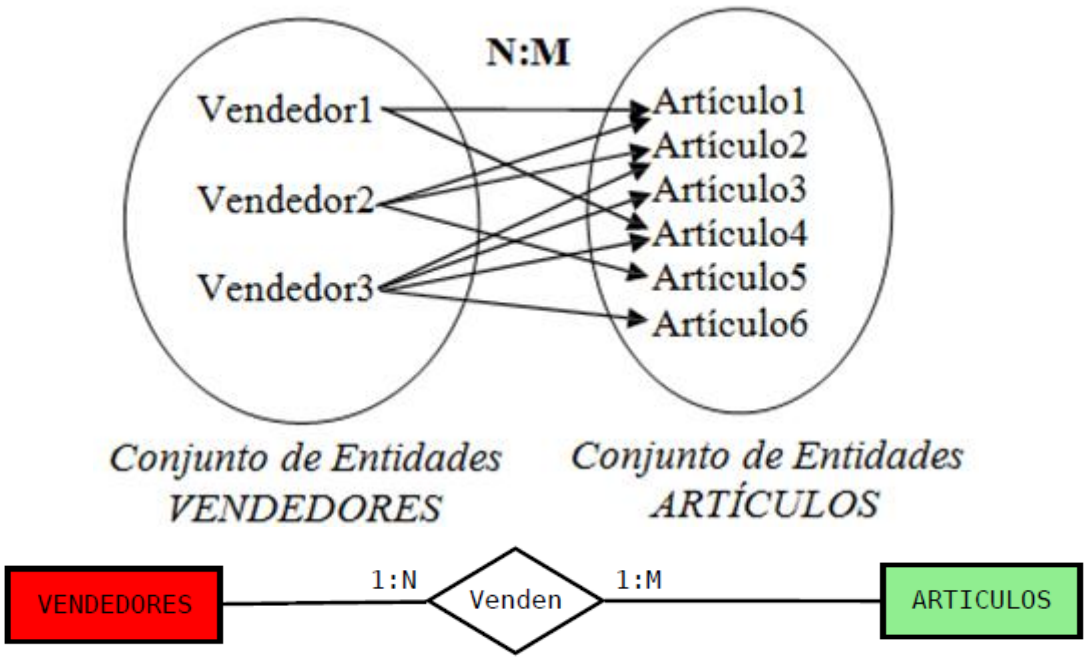
2. **1:N o uno a muchos**: A cada elemento de la primera entidad le corresponde uno o más elementos de la segunda entidad, y a cada elemento de la segunda entidad le corresponde uno sólo de la primera entidad. Por ejemplo, un mismo proveedor suministra varios artículos a una empresa, y cada artículo que adquiere la empresa siempre es pedido al mismo proveedor.





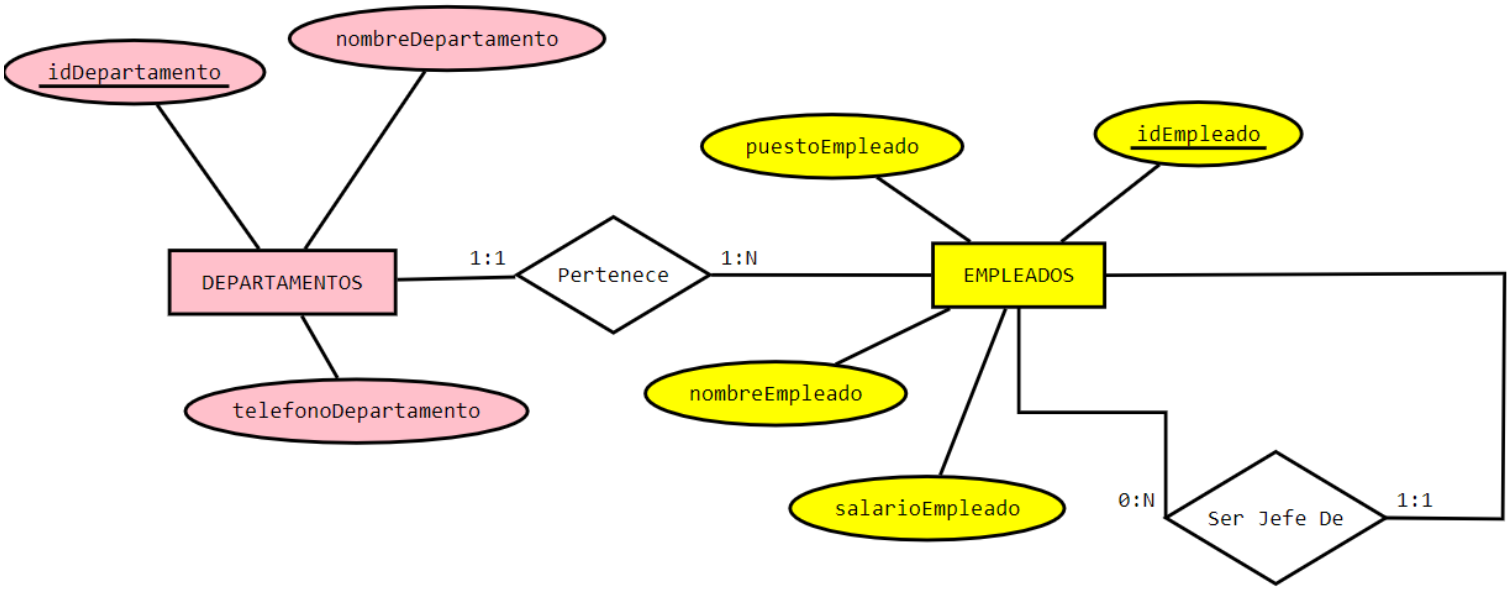
3. **N:M o muchos a muchos**: A cada elemento de la primera entidad le corresponde uno o más elementos de la segunda entidad, y a cada elemento de la segunda entidad le corresponde uno o más elementos de la primera entidad. Por

ejemplo, cada vendedor de una tienda vende muchos artículos y cada artículo es vendido por varios vendedores.



La **cardinalidad** de una entidad informa del grado de participación de dicha entidad concreta en la relación. Se expresan entre paréntesis indicando los valores máximo y mínimo. Los valores son: (0,1), (1,1), (0,N), (1,N) y (N,M). El valor 0 se pone cuando la participación de la entidad es opcional.

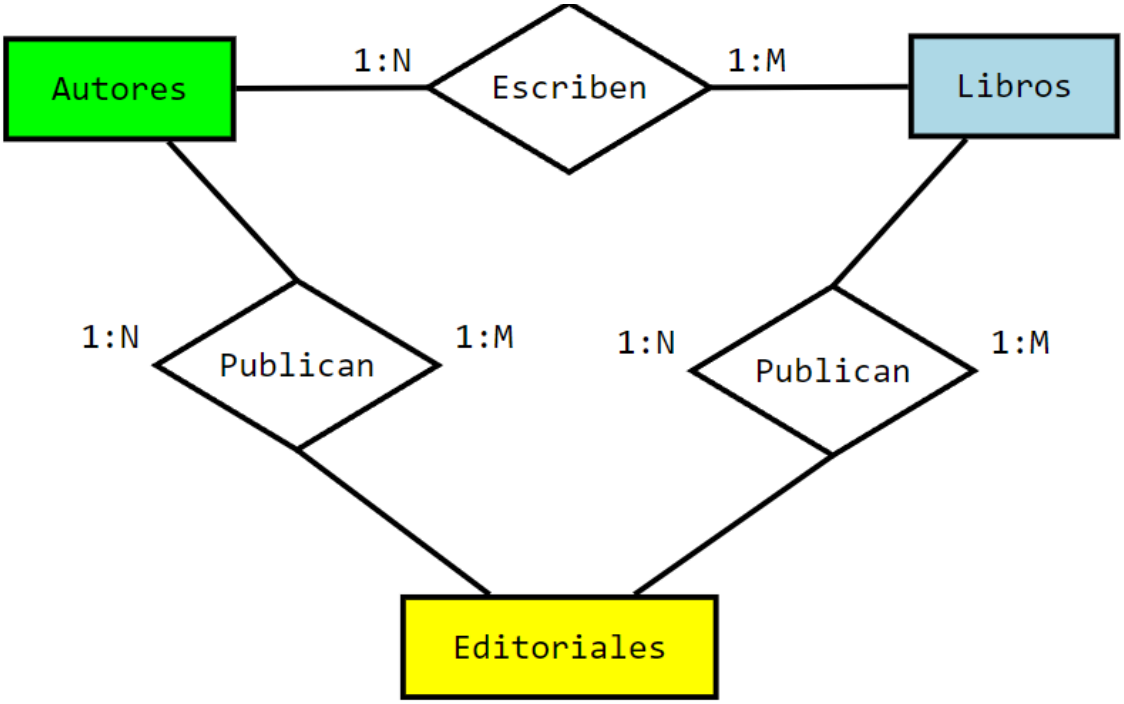
El ejemplo completo del diagrama E-R en el que se relacionaban las entidades EMPLEADOS y DEPARTAMENTOS sería:



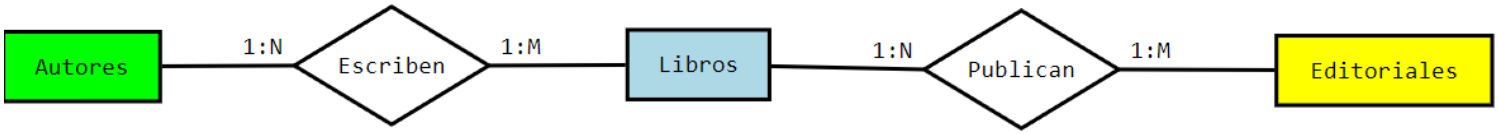
A cada departamento le pertenece 1 o más empleados. Cada empleado pertenece a un departamento y sólo a uno. Un empleado tiene un jefe y sólo uno. Un empleado que es jefe, lo es de uno o más empleados.

Redundancias

En muchas ocasiones tenemos dudas a la hora de relacionar varias entidades entre sí. Por ejemplo, en el caso de los Autores que escriben Libros y que publican con Editoriales podríamos tener una primera aproximación así:



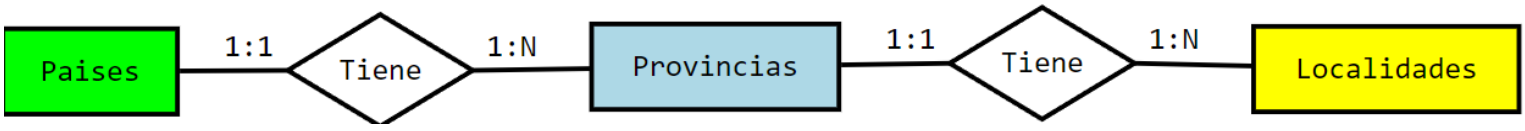
Pero si lo estudiamos con detenimiento, podemos llegar a la conclusión siguiente:



Más sencillo y donde podemos guardar la misma información, pero más eficientemente pues si sabemos que un libro es escrito por un autor y que dicho libro se ha editado por tal editorial, ya tenemos implícitamente la relación entre Autores y Editoriales.

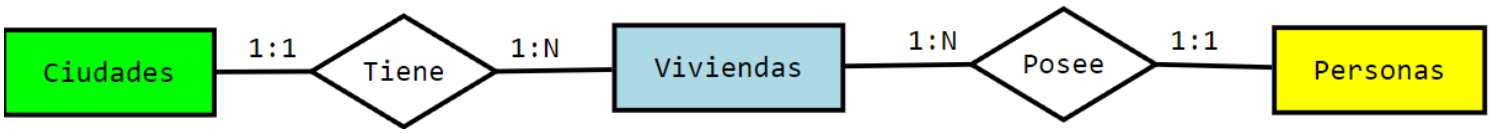
Esto no siempre es posible, hay que estudiarlo con cuidado.

Otro ejemplo, en la relación entre Localidades, Provincias y Países:



Aquí, por las cardinalidades, para cualquier Localidad, podemos averiguar fácilmente a qué País pertenece. De Localidades a Países vamos siempre encontrando 1:1 en la cardinalidad izquierda.

Sin embargo, en la siguiente relación entre Ciudades, Viviendas ubicadas en ellas y las Personas dueñas de estas:



Aquí, por las cardinalidades y el sentido de estas, dada una persona, no podríamos saber en qué Ciudad vive, pues podría darse el caso de tener varias viviendas en varias ciudades. Tenemos información, sí, pero no tan precisa como en el caso anterior.

Generalización y Jerarquías de Generalización

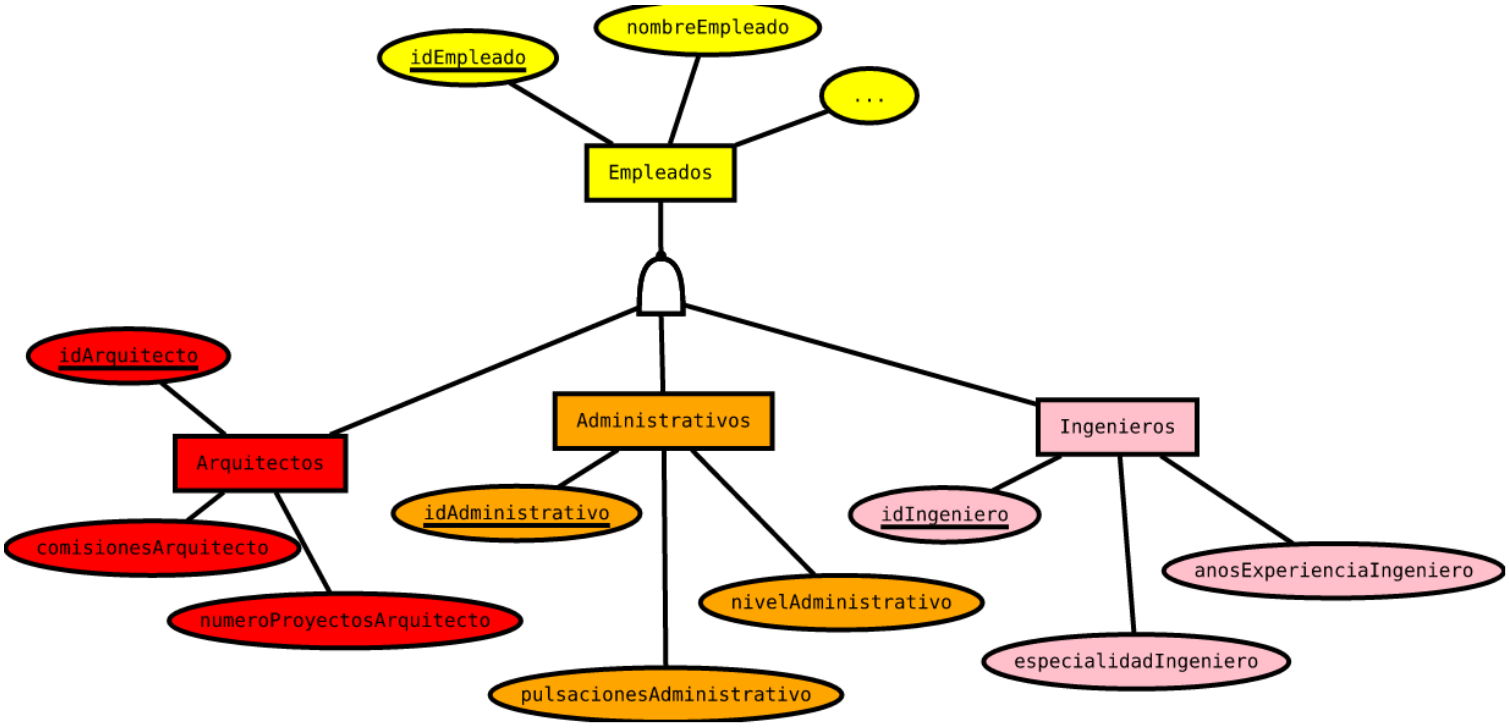
Las **generalizaciones** proporcionan un mecanismo de abstracción que permite especializar una entidad, denominada supertipo, en subtipos. También se dice que se generalizan los subtipos en el supertipo.

Una generalización se identifica si encontramos una serie de **atributos comunes** a un conjunto de entidades, y unos **atributos específicos** que identificarán unas características. Los atributos comunes describirán el supertipo y los particulares los subtipos.

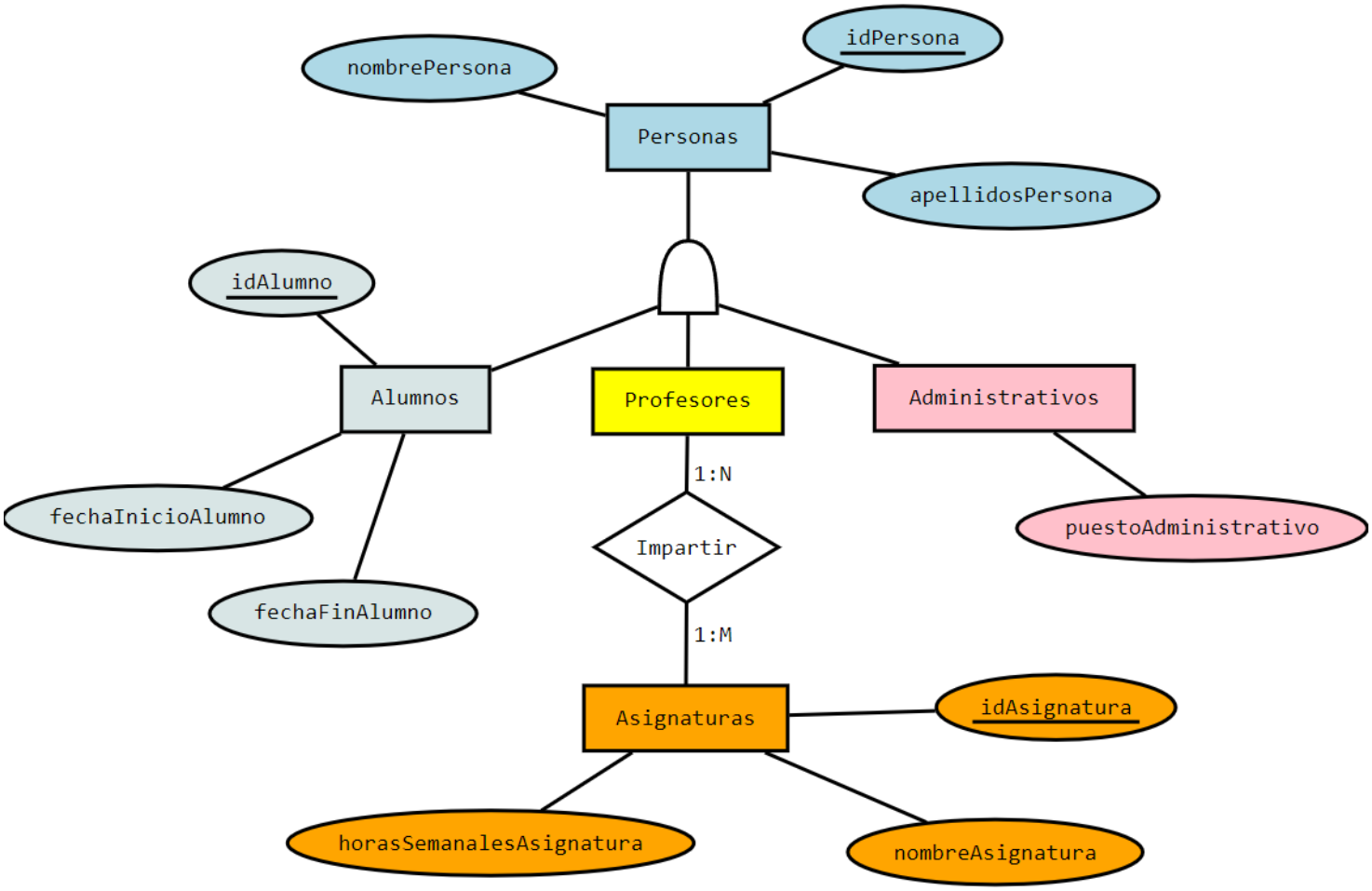
Por ejemplo, en una empresa de construcción se identifican las siguientes entidades:

- Empleados (**idEmpleado**, nombreEmpleado, direccionEmpleado, fechaNacimientoEmpleado, salarioEmpleado, puestoEmpleado)
  - Arquitectos, que incluye los atributos de un empleado más los atributos específicos: numeroProyectosArquitecto y comisionesArquitecto.
  - Administrativos, que incluye los atributos de un empleado más los atributos específicos: pulsacionesAdministrativo y departamentoAdministrativo.
  - Ingenieros, que incluye los atributos de un empleado más los atributos específicos: especialidadIngeniero y anosExperienciaIngeniero.

La **herencia** es el mecanismo por el que los atributos del supertipo son "heredados" por sus subtipos.



Otro ejemplo:



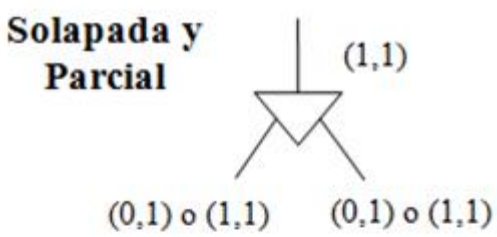
www.grupostudium.com
informacion@grupostudium.com
954 539 952

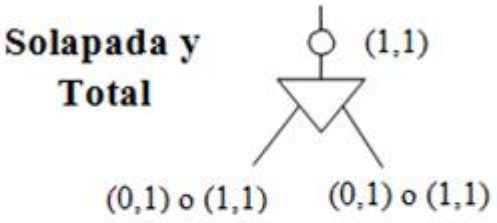
La generalización es **total** si no hay ocurrencias en el supertipo que no pertenezcan a ninguno de sus subtipos, es decir, los empleados de la empresa, o son arquitectos, o administrativos, o ingenieros, no hay de otro tipo.

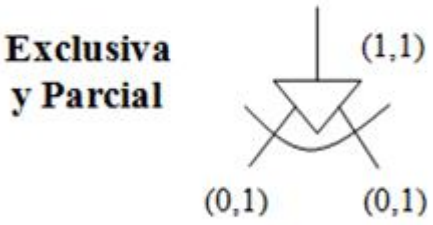
La generalización es **parcial** si hay empleados que no pertenecen a ningún subtipo, es decir, que exista algún empleado que no sea ni arquitecto, ni administrativo, ni ingeniero.

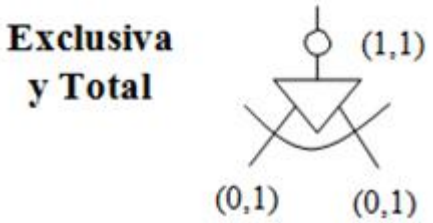
La generalización es **exclusiva** si cada empleado sólo puede pertenecer a un único subtipo, y no a más de uno. Si un empleado puede ser varias cosas a la vez la generalización es **solapada** o **superpuesta**.

Nosotros, por simplificación, y por ser lo más común, plantearemos siempre la Exclusiva y Total. En caso contrario, ya se indicará convenientemente.









www.grupostudium.com
informacion@grupostudium.com
954 539 952

Ejemplos

Veamos algunos ejemplos de lo visto hasta ahora.

Ejemplo 1: Supongamos que en un centro escolar se imparten muchos cursos. Cada curso está formado por un grupo de alumnos, de los cuales uno de ellos es el delegado del grupo. Los alumnos cursan asignaturas, y una asignatura puede o no ser cursada por los alumnos. Vamos a representar este supuesto en un ERD paso a paso.

Primero vamos a identificar las entidades, luego las relaciones y las cardinalidades y, por último, los atributos de las entidades y de las relaciones, si las hubiera.

Como entidades podemos pensar en un primer momento en CENTROS, CURSOS, ALUMNOS, ASIGNATURAS Y DELEGADOS. Si analizamos CENTROS veremos que se puede descartar ya que se trata del propio centro y es único. Si se tratara de una gestión de varios centros, tendría más sentido. Por otro lado, los DELEGADOS son ALUMNOS, así que también se puede descartar. Nos quedaríamos con CURSOS, ALUMNOS y ASIGNATURAS.

Del propio enunciado sacamos la relaciones: un curso está formado por muchos alumnos (1:N), de los alumnos que pertenecen a un grupo, uno es delegado de muchos (1:N), siendo además una relación reflexiva, y por último, muchos alumnos pueden cursar muchos cursos (N:M)

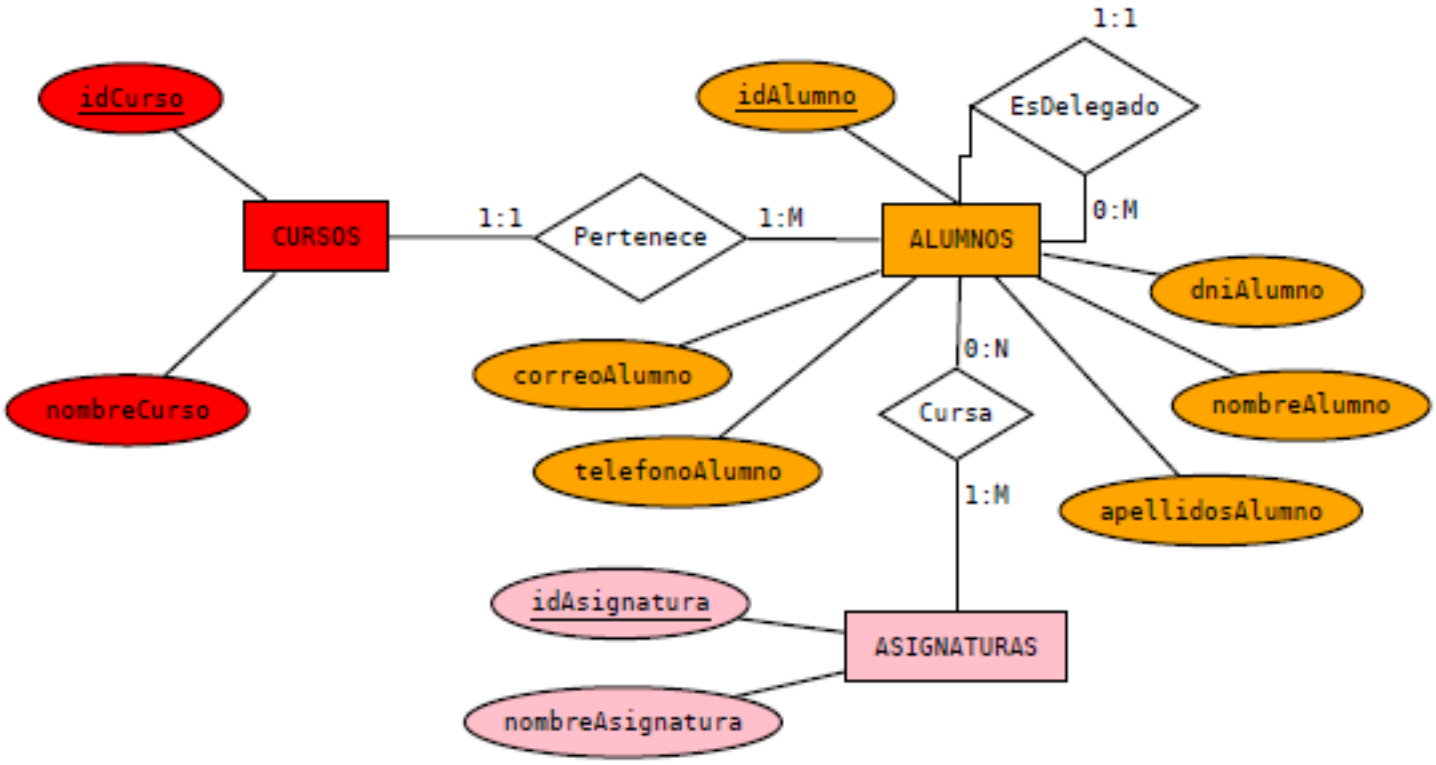
Como atributos tendremos los siguientes que asignamos con un poco de sentido común, pues en el enunciado no se nos indica ninguno. Como mínimo un identificador que será el campo clave y un nombre o descripción.

CURSOS: idCurso y nombreCurso

ALUMNOS: idAlumno, dniAlumno, nombreAlumno, apellidosAlumno, telefonoAlumno y correoAlumno

ASIGNATURAS: idAsignatura y nombreAsignatura

Por último, representamos el diagrama:



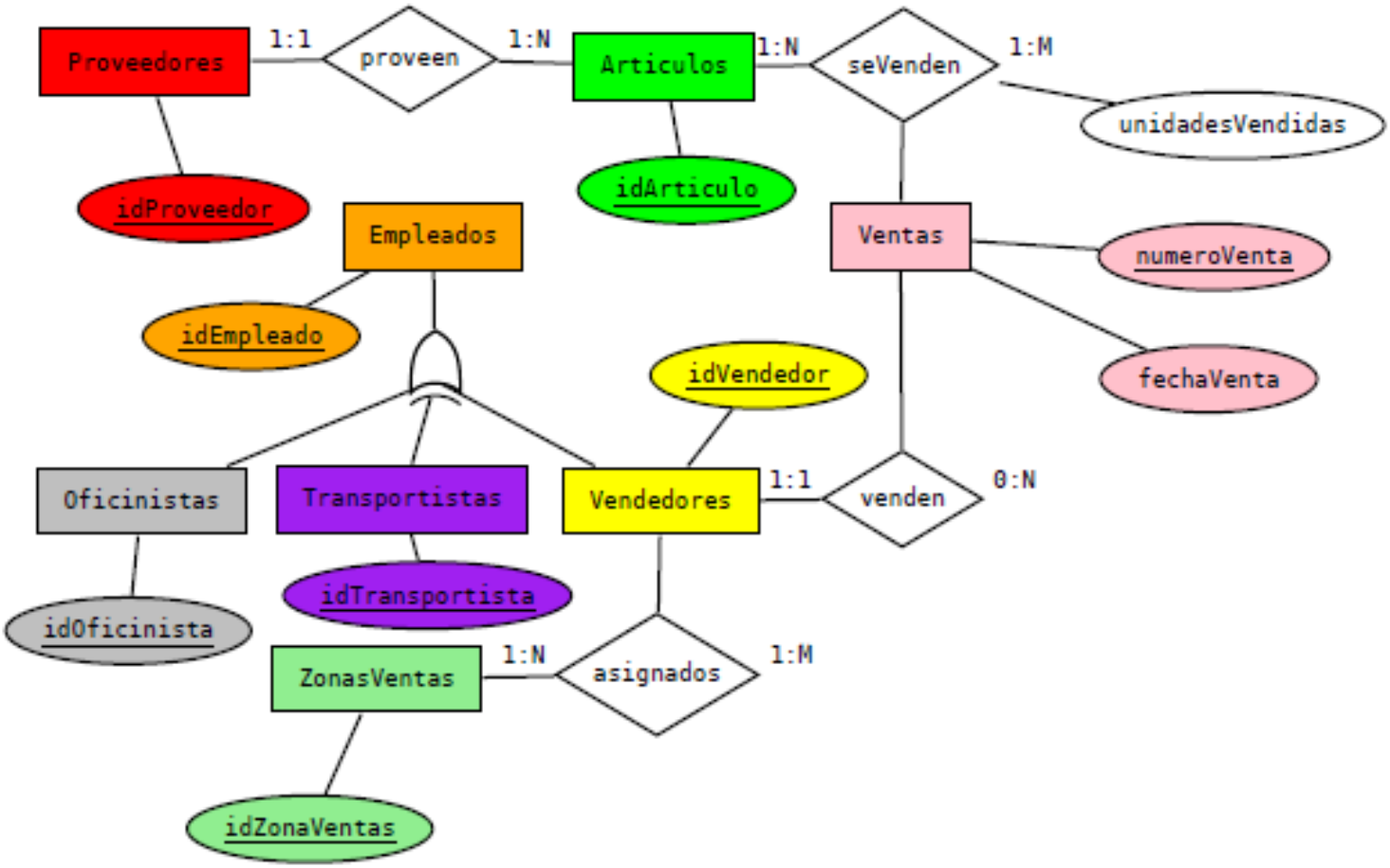
Ejemplo 2: Indicar el tipo de generalización de los siguientes supuestos.

Supuesto 1: Un concesionario de coches vende coches nuevos y usados. Los atributos específicos de los nuevos son las unidades y el descuento; de los usados son los kilómetros y el año de fabricación. **Exclusiva Total**

Supuesto 2: Consideramos el conjunto de personas de una ciudad, distinguimos a los trabajadores, estudiantes y parados. De los trabajadores nos interesa el número de la Seguridad Social, la empresa de trabajo y el salario. De los estudiantes, el número de matrícula y el centro educativo, y de los parados la fecha del paro. **Solapada Total**, pues una persona puede ser trabajadora y estudiante o estudiante y parado.

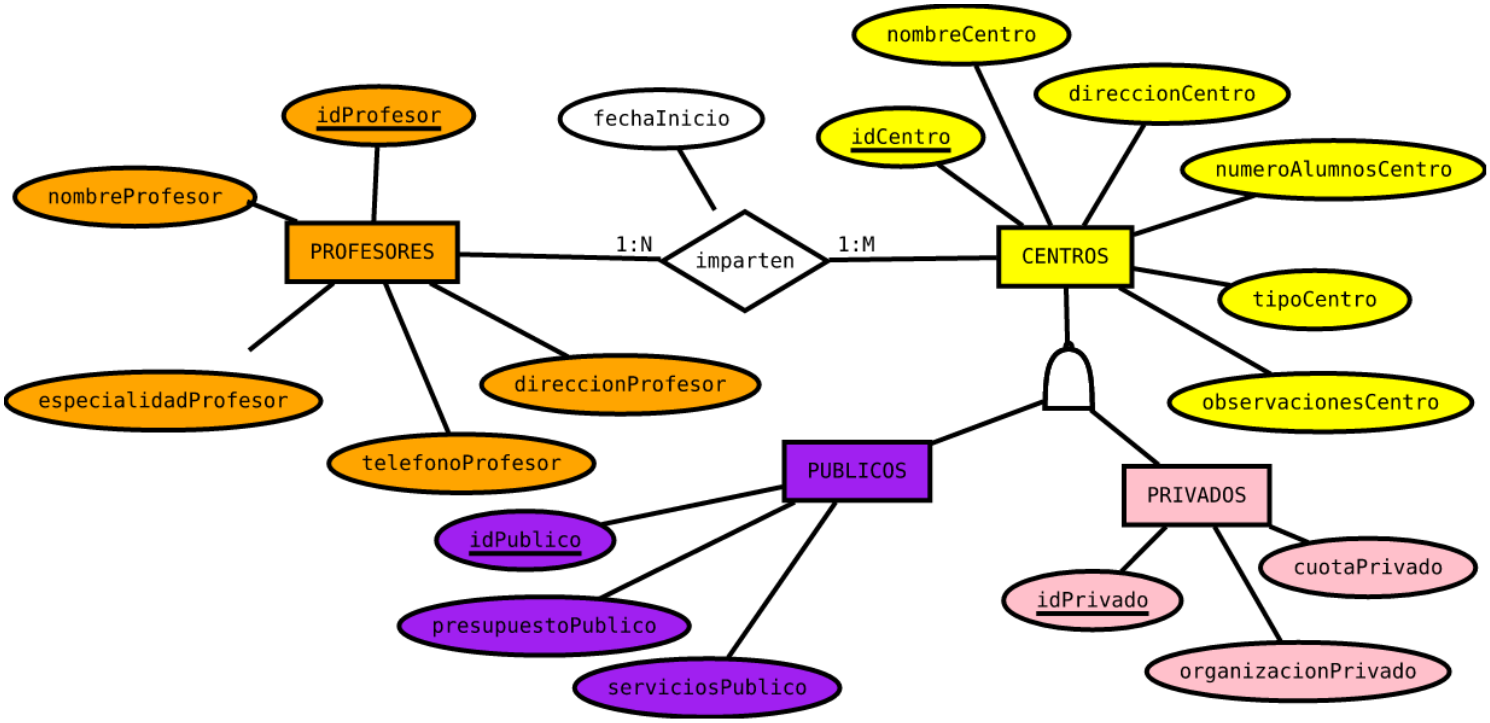
Supuesto 3: En un campo de fútbol los puestos de los futbolistas pueden ser portero, defensa, medio y delantero. **Exclusiva Total**

Ejemplo 3: Una compañía de distribución de productos para el hogar dispone de proveedores que le suministran artículos. Un artículo sólo puede proveerlo un proveedor. La empresa tiene tres tipos de empleados: oficinistas, transportistas y vendedores. Estos últimos vende los artículos. Una venta la realiza un único vendedor, y un vendedor puede vender en distintas zonas de venta. De las ventas nos interesa saber la fecha de venta y las unidades vendidas.



NOTA: Se trata de una generalización de tipo Exclusiva Total.

Ejemplo 4: Hay profesores que imparten clases en 2 tipos de centros educativos: públicos y privados. Un profesor puede impartir clase en varios centros, ya sean públicos o privados. La asignatura será un atributo de la relación entre el profesor y el centro donde imparte. Los centros educativos sólo pueden ser públicos o privados. Un centro público no puede ser privado a la vez, ni a la inversa. Los atributos específicos para los centros públicos son: el presupuesto y los servicios; y para los privados son: la organización y la cuota.



2.4 El Modelo Relacional

El **modelo relacional** fue desarrollado por **E. F. Codd** para IBM a finales de los años sesenta, y mantiene la independencia de la estructura lógica de la base de datos respecto al modo de almacenamiento y otras características de tipo físico.
El modelo relacional persigue, al igual que la mayoría de los modelos de datos, los siguientes objetivos:

1. **Independencia física de los datos**: El modo de almacenamiento de los datos no debe influir en su manipulación lógica.
2. **Independencia lógica de los datos**: Los cambios que se realicen en los objetos de la base de datos sin alterar los datos almacenados previamente no deben repercutir en los programas y usuarios que acceden a la misma.
3. **Flexibilidad**: Para representar a los usuarios los datos de la forma más adecuada a la aplicación que utilicen.
4. **Uniformidad**: En la presentación de las estructuras lógicas de los datos, que son tablas, lo que facilita la concepción y manipulación de la base de datos por parte de los usuarios.
5. **Sencillez**: Pues las características anteriores, así como unos lenguajes de usuario sencillos, hacen que este modelo sea fácil de comprender y utilizar por el usuario.

Codd introduce el concepto de relación (**tabla**) como estructura básica del modelo. Todos los datos de una Base de Datos se representan en forma de relaciones cuyo contenido varía en el tiempo. El modelo relacional se basa en 2 ramas de las matemáticas: la teoría de conjuntos y la lógica de predicados.

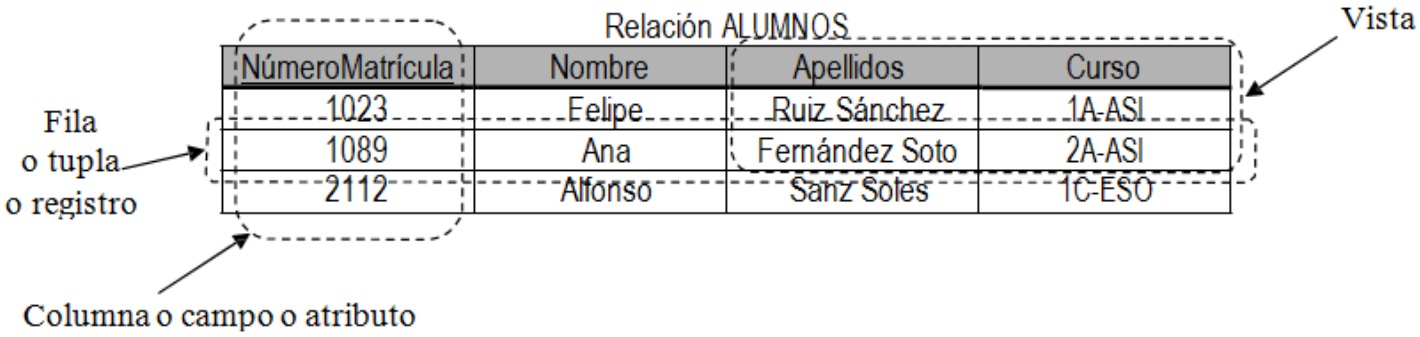
Relación EMPLEADOS

Numero_Empleado	Nombre_Completo	Numero_Departamento	Fecha_Alta
154621	Andrés Pérez Ruiz	15	10/08/2001
889701	Nuria Gálvez Pinar	8	06/05/2002

Ejemplo de una relación (tabla) con 4 campos (columnas o atributos) y 2 tuplas (filas o registros).

Estructura del Modelo de Datos Relacional

La **relación** es el elemento básico del modelo relacional, y se representa como una tabla, con su nombre y las columnas que representan los atributos. A las filas de la tabla se las llama **tuplas**, y contienen los valores que toman cada uno de los atributos para cada elemento de la relación. Una representación de una relación en forma de tabla podría ser:



NúmeroMatrícula	Nombre	Apellidos	Curso
1023	Felipe	Ruiz Sánchez	1A-ASI
1089	Ana	Fernández Soto	2A-ASI
2112	Alfonso	Sanz Soles	1C-ESO

Una **relación** (o tabla) está formada por:

1. **Atributo** (o campo): Cada columna de la tabla. Tiene su propio nombre y pueden guardar un conjunto de valores. El orden de las columnas en una tabla es irrelevante, una columna se identifica por su nombre no por su posición.
2. **Tupla** (o registro): Cada fila de la tabla. De las tablas se derivan los siguientes conceptos:
   - Cardinalidad: Número de filas de la tabla. La relación ALUMNOS tenía cardinalidad 3.
   - Grado: Número de columnas de la tabla. La relación ALUMNOS tenía grado 4.
   - Valor: Intersección entre una fila y una columna. Un valor de la relación ALUMNOS puede ser 1089.
   - Valor Null (o valor nulo): Representa la ausencia de información.

Propiedades de las relaciones:

1. No puede haber 2 relaciones con el mismo nombre en la Base de Datos.
2. En cada tupla (fila), cada atributo toma un único valor (tienen valores atómicos). Se dice que las relaciones están normalizadas[1].
3. No puede haber 2 atributos con el mismo nombre en una relación.
4. No puede haber 2 tuplas iguales en una relación.
5. Al igual que con las columnas, el orden de las filas es irrelevante, es decir, las tuplas no están ordenadas.

[1] La normalización se verá posteriormente

Tipos de relaciones más importantes:

1. **Relaciones base**: Son relaciones reales que tienen nombre y forman parte directa de la Base de Datos almacenada. Se corresponden con el nivel conceptual de la arquitectura ANSI.
2. **Vistas** (o relaciones virtuales): Son relaciones con nombre que se definen a partir de una consulta, no tienen datos almacenados, lo único que almacena es la definición de la consulta a realizar. Se corresponden con el nivel de visión o externo de la arquitectura ANSI.

Una **clave** permite identificar a una fila de una tabla. En una tabla no hay filas repetidas, se identifican de un modo único mediante los valores de sus atributos. A veces la clave está formada por un único atributo, pero otras veces debe formarse por más de un atributo. Una clave debe cumplir 2 requisitos:

1. En cada fila de la tabla, el valor de la clave ha de identificarla de forma unívoca.
2. No se puede descartar ningún atributo de la clave para identificar la fila.

Una **clave primaria** o **clave principal** (Primary key) es aquella clave candidata que el diseñador escoge para identificar las tuplas de la relación. No puede tener valores nulos.

Una **clave ajena** (Foreign key) de una tabla es el conjunto de atributos cuyos valores han de coincidir con los valores de la clave primaria de otra tabla. A través de las claves ajenas se construyen asociaciones entre tablas. Ambas claves deberán estar definidas sobre el mismo dominio y son muy importantes en el estudio de la integridad de datos del modelo relacional.

Esquema de una Base de Datos Relacional

Una **base de datos relacional** es un conjunto de relaciones normalizadas. Para representar su esquema deben darse:

1. El nombre de sus **relaciones**.
2. Los **atributos** de las relaciones.
3. Las claves primarias y ajenas.

Las claves ajenas o foráneas se representan mediante diagramas referenciales.

Tabla DEPARTAMENTOS

CódigoDpto	NombreDpto	Presupuesto
D1	Marketing	1000,00
D2	Investigación	2500,50
D3	Desarrollo	5200,00

Tabla EMPLEADOS

IdEmpleado	NombreCompleto	Departamento	Salario
E1	Luis Pérez Sánchez	D2	1500,00
E2	Ana Vázquez Molino	D1	1500,00
E3	Paula Sanz Pino	D2	1650,00
E4	Felipe Ruiz Ferrer	D3	1450,00

El esquema completo de esta base de datos definiendo Tablas, Atributos y Claves primarias (atributos subrayados) sería:

DEPARTAMENTOS (CodigoDpto, NombreDpto, Presupuesto)
EMPLEADOS (IdEmpleado, NombreCompleto, Departamento, Salario)

Restricciones del Modelo Relacional

El modelo relacional impone 2 tipos de restricciones para tener en cuenta a la hora de diseñar una base de datos:

1. **Restricciones inherentes al modelo relacional (algunas ya vistas como propiedades de las relaciones):**
   - 1.1. En una relación no puede haber 2 tuplas iguales.
   - 1.2. El orden de las tuplas y de los atributos es irrelevante.
   - 1.3. Cada atributo de cada tupla sólo tiene un valor perteneciente al dominio al que corresponde.
   - 1.4. Ningún atributo que forme parte de la clave primaria puede tomar valor nulo.

2. **Restricciones semánticas o de usuario:**

2.1. Restricción de la clave primaria (PRIMARY KEY en SQL): permite al usuario declarar uno o varios atributos como clave primaria en una relación.

2.2. Restricción de unicidad (UNIQUE en SQL): permite al usuario definir claves alternativas, pues los atributos marcados como únicos no pueden repetirse.

2.3. Restricción de obligatoriedad (NOT NULL en SQL): permite al usuario declarar si uno o varios atributos no pueden tomar valores nulos, por tanto, deben tener siempre un valor.

2.4. Restricción de clave ajena (FOREIGN KEY en SQL) o integridad referencial: El usuario la utiliza para asociar relaciones de una Base de Datos mediante claves ajenas. La integridad referencial indica que los valores de la clave ajena de esa relación se corresponden obligatoriamente con los valores de la clave primaria de otra relación. Tras definir las claves ajenas hay que tener en cuenta qué ocurrirá cuando se borren (ON DELETE en SQL) o se modifiquen (ON UPDATE en SQL) tuplas de la relación cuya clave primaria se corresponde con esta clave ajena.

2.4.1. Borrado y/o modificación en cascada (CASCADE en SQL): Si el usuario especifica esta restricción, el borrado o modificación de una tupla en una relación ocasiona un borrado o modificación de las tuplas de otras relaciones asociadas a ésta, cuyas claves ajenas se correspondan con su clave primaria. Por ejemplo, si se borra un departamento de la tabla DEPARTAMENTOS, se borrarán todos los empleados de la tabla EMPLEADOS que pertenezcan a ese departamento borrado; y si se modifica el código de un departamento de la relación DEPARTAMENTOS, automáticamente cambiarán todos los valores del campo Departamento en la relación EMPLEADOS que tenía el mismo código al nuevo valor ya modificado.

2.4.2. Borrado y/o modificación restringidos (RESTRICT en SQL): Si el usuario especifica esta restricción, el borrado o modificación de una tupla en una relación será imposible si existen tuplas de otras relaciones asociadas a ésta, cuyas claves ajenas se correspondan con su clave primaria. Así, no se podría, por ejemplo, eliminar un departamento si tiene empleados asociados a él, ni tampoco se podría cambiar su código si ya tenía empleados; por tanto, únicamente podrán realizarse esas operaciones en departamentos sin empleados asociados.

2.4.3. Borrado y/o modificación con puesta a nulos (SET NULL en SQL): Si el usuario especifica esta restricción, el borrado o modificación de una tupla en una relación ocasiona la puesta a NULL de la clave ajena de las tuplas de otras relaciones asociadas a ésta, cuyas claves ajenas se correspondían con su clave primaria.

2.4.4. Borrado y/o modificación con puesta a valor por defecto (SET DEFAULT en SQL): Si el usuario especifica esta restricción, el borrado o modificación de una tupla en una relación ocasiona la puesta al valor por defecto especificado de la clave ajena de las tuplas de otras relaciones asociadas a ésta, cuyas claves ajenas se correspondían con su clave primaria.

2.5. Restricción de verificación (CHECK en SQL): Esta restricción permite al usuario especificar condiciones que deban cumplir los valores de los atributos. Cada vez que se realiza una inserción o una actualización de datos se comprueba si los valores cumplen la condición, rechazando la operación si no la cumplen.

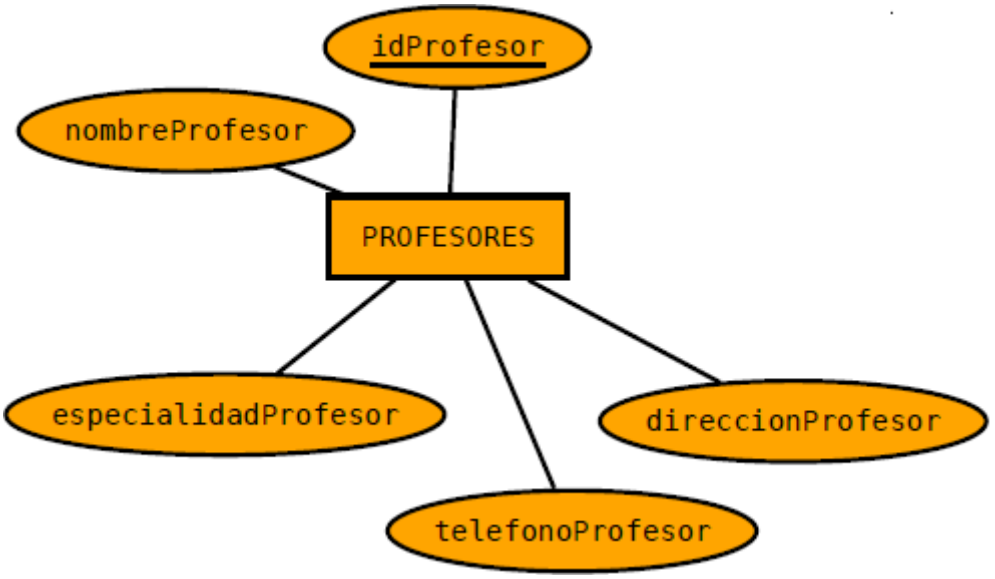
2.6. Aserciones (ASSERTION en SQL): Igual que CHECK, pero puede afectar a 2 o más relaciones, por tanto, la condición a cumplir se establece sobre campos de distintas relaciones. Pueden implicar a subconsultas en la condición.

2.5 Transformación de un Diagrama Entidad-Relación a un Esquema Relacional

Una vez obtenido el esquema conceptual mediante un diagrama E-R, puede definirse el modelo lógico de datos mediante un esquema relacional.

Las reglas básicas para transformar un diagrama E-R a un esquema relacional son:

1. Toda **entidad** se transforma en una tabla.
2. Todo **atributo** de la entidad se transforma en columna o campo de la tabla.
3. La **clave primaria** de la entidad se transforma en la clave primaria de la tabla.



De este diagrama E-R sacaríamos el siguiente esquema relacional:

PROFESORES (**idProfesor**, nombreProfesor, especilidadProfesor, telefonoProfesor, direccionProfesor)

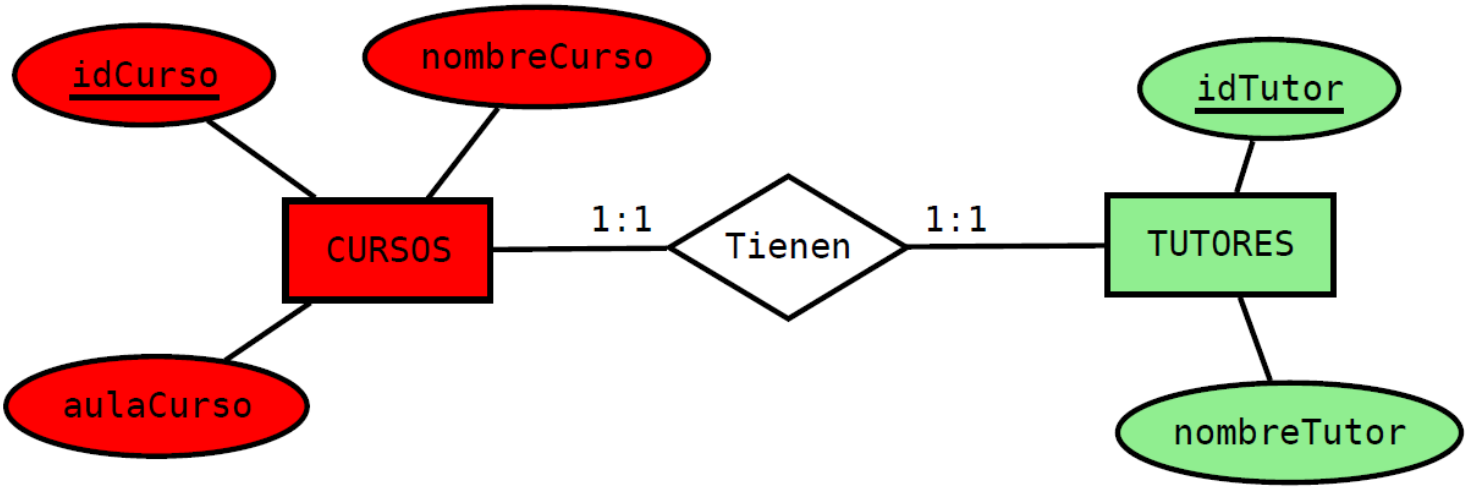
Podemos identificar la clave primaria mediante subrayado, **negrita**, un asterisco (*), o cualquier otro elemento que pueda diferenciar a dicho campo clave de los demás.

4. Para las relaciones 1:1 se tienen en cuenta las cardinalidades de las entidades que participan en la relación.

   4.1. Unir ambas entidades en una tabla, cuando ambas entidades tienen cardinalidad (1,1). Se escoge como clave primaria de la tabla a una cualquiera de las dos claves primarias de las entidades.

   4.2. Propagar la clave, cuando una entidad tiene cardinalidad (1,1) y la otra (0,1). Se propaga la clave primaria de la entidad con cardinalidad (1,1) a la tabla resultante de la entidad con cardinalidad (0,1) convirtiéndose en clave ajena.

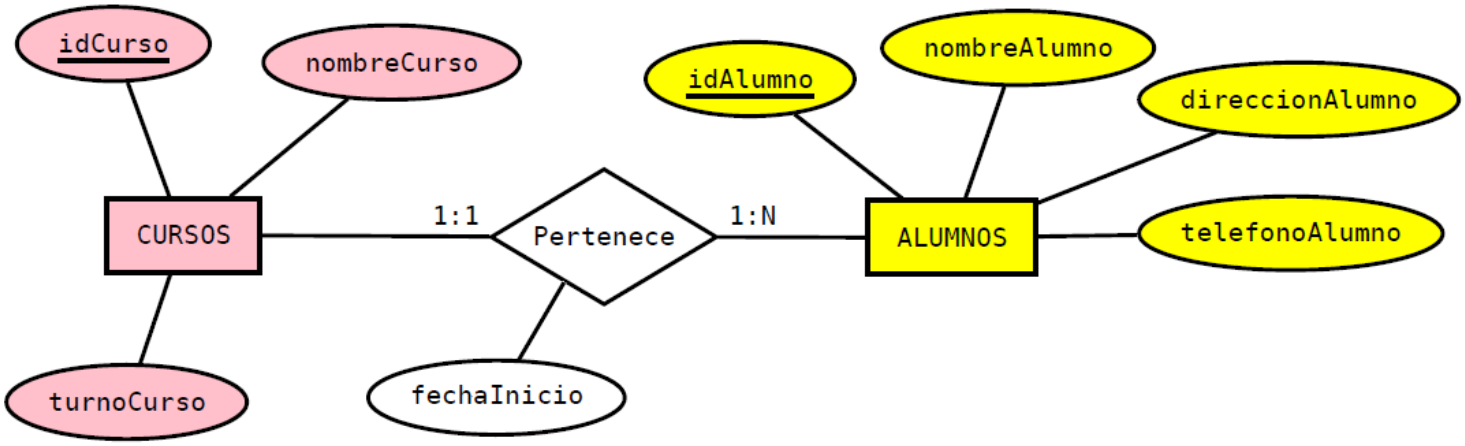
   4.3. Transformar la relación en una tabla, cuando ambas entidades tienen cardinalidad (0,1). Se transforma la relación en una tabla independiente, tal y como se hacía cuando la relación era N: M.



Esto daría como esquema relacional:

TUTORES_CURSOS (**idCurso**, nombreCurso, aulaCurso, idTutor, nombreTutor)

5. Para las relaciones 1:N se propaga la **clave**, es decir, se propagan, por un lado la clave primaria de la entidad con cardinalidad máxima 1 y por otro, los atributos de la propia relación a la entidad cuya cardinalidad máxima es N.



El esquema relacional quedaría de la siguiente manera:

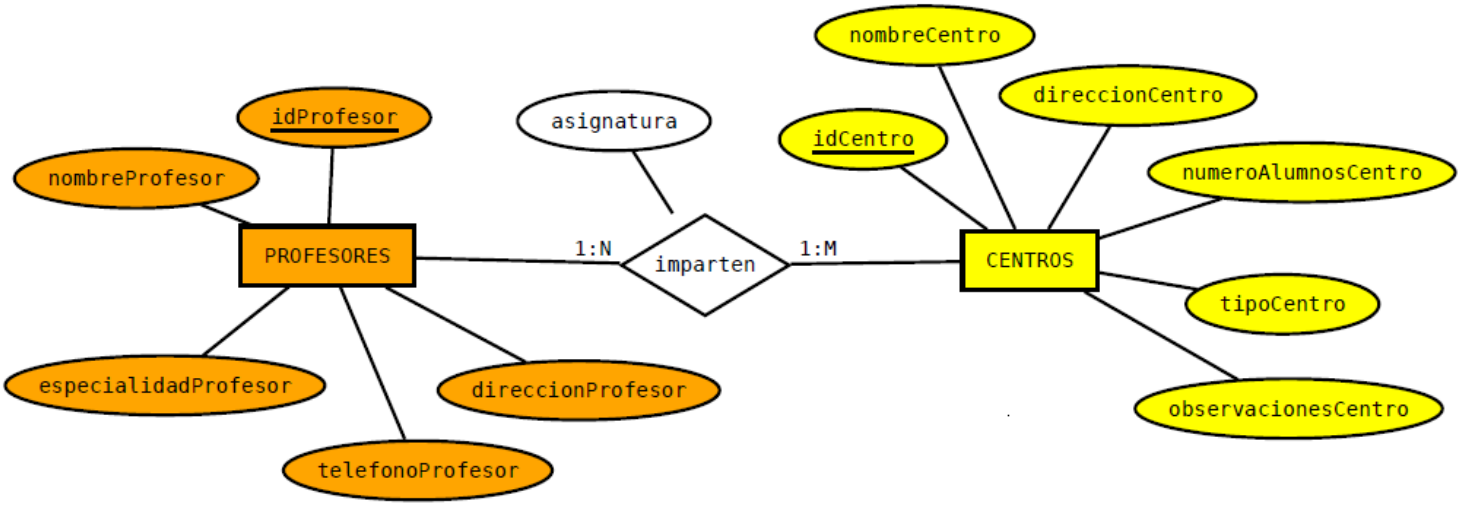
CURSOS (**idCurso**, nombreCurso, turnoCurso
ALUMNOS (**idAlumno**, nombreAlumno, direccionAlumno, telefonoAlumno

Si nos fijamos, aún no hemos cerrados los paréntesis. Eso se debe a que nos falta la información de la relación y de su atributo. Se tratarían de Foreign Keys, con lo que agregaremos FK al final del atributo correspondiente. Hay dos posibilidades: idCursoFK y fechaInicio en ALUMNOS o idAlumnoFK y fechaInicio en CURSOS. Esta última opción no es válida pues si a un CURSO pertenecen muchos alumnos, el idAlumnoFK de qué alumno ponemos si hay muchos. Por tanto, lo correcto sería a cada alumno ponerle el idCursoFK al que pertenece.

De esta forma el esquema relacional quedaría completo así:

CURSOS (**idCurso**, nombreCurso, turnoCurso)
ALUMNOS (**idAlumno**, nombreAlumno, direccionAlumno, telefonoAlumno, idCursoFK, fechaInicio)

6. Toda **relación N:M** se transforma en una **tabla**, que tendrá como claves ajenas las claves primarias de las entidades que asocia. Y podrá tener como clave primaria la concatenación de los atributos clave de las entidades que asocia si es posible, si no, se utilizan junto con uno o varios atributos de la relación o se le agrega un campo identificador nuevo como clave primaria.



www.grupostudium.com
informacion@grupostudium.com
954 539 952

De este diagrama E-R sacaríamos el siguiente esquema relacional:

Tablas que salen directamente de Entidades:

PROFESORES (**idProfesor**, nombreProfesor, especilidadProfesor, telefonoProfesor, direccionProfesor)
CENTROS (**idCentro**, nombreCentro, direccionCentro, numeroAlumnosCentro, tipoCentro, observacionesCentro)

Tabla que sale de la relación N:M:

IMPARTICIONES (**idProfesorFK**, **idCentroFK**, asignatura)

En este caso hemos cogido como campos clave los campos clave de las entidades que asocia, pero también podríamos haber puesto:

IMPARTICIONES (**idImparticion**, idProfesorFK, idCentroFK, asignatura)

Esta segunda forma es menos habitual. Y no son equivalentes. En algunas ocasiones no se pueden usar **ambas**. La segunda forma (nuevo campo clave) SIEMPRE es correcta. En este caso, si un profesor dar más de una asignatura en un centro, la primera solución sería incorrecta. Pues la pareja idProfesorFK-idCentroFK no se puede repetir por ser el **campo clave**. En este caso la opción correcta sería:

IMPARTICIONES (**idImparticion**, idProfesorFK, idCentroFK, asignatura)

La otra solución sería incluir asignatura como campo clave:

IMPARTICIONES (**idProfesorFK**, **idCentroFK**, **asignatura**)

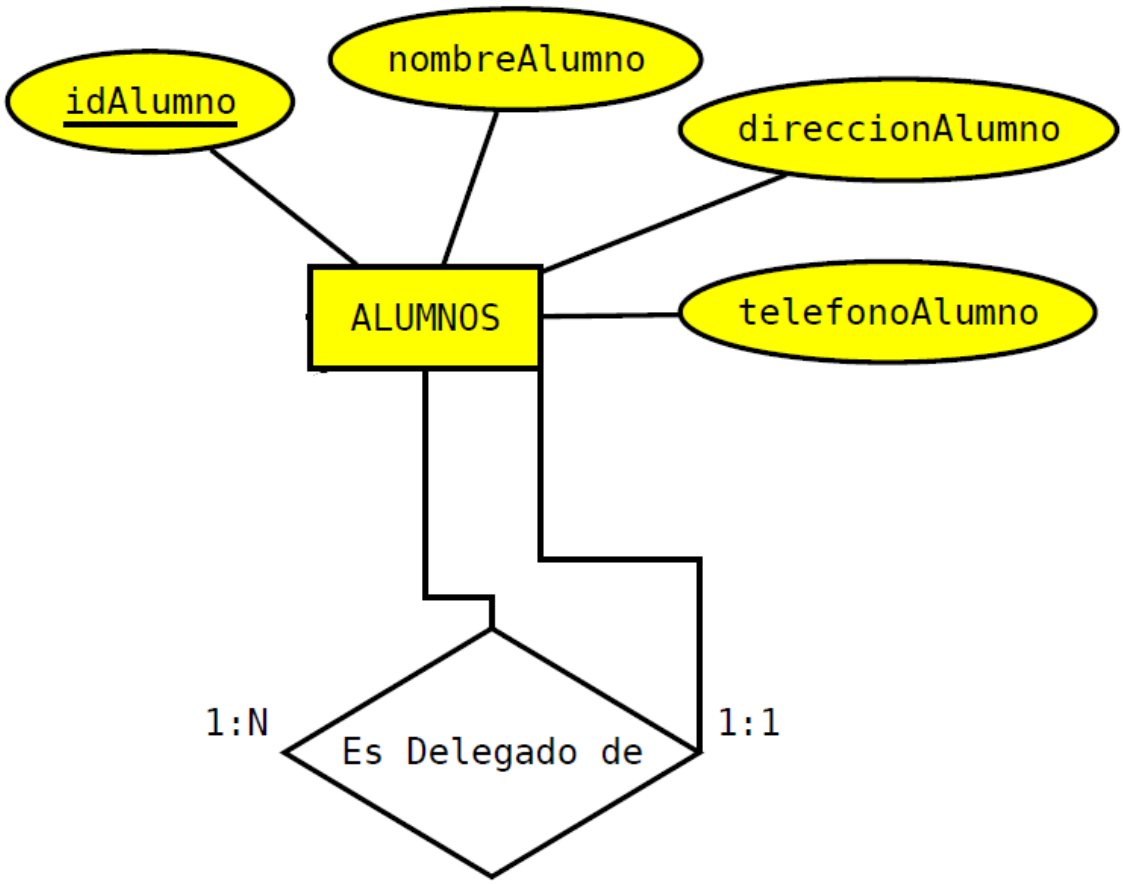
Pero esto es menos eficiente.

Transformación de otros elementos del modelo E-R:

1. **Relaciones reflexivas**: Son las relaciones binarias en las que únicamente participa un tipo de entidad. Pueden encontrarse los siguientes casos:

   1.1. Si la relación es 1:1, no se crea una segunda tabla, si no que en la tabla resultante se agregará 2 veces el mismo atributo, como clave primaria y como clave ajena a ella misma.

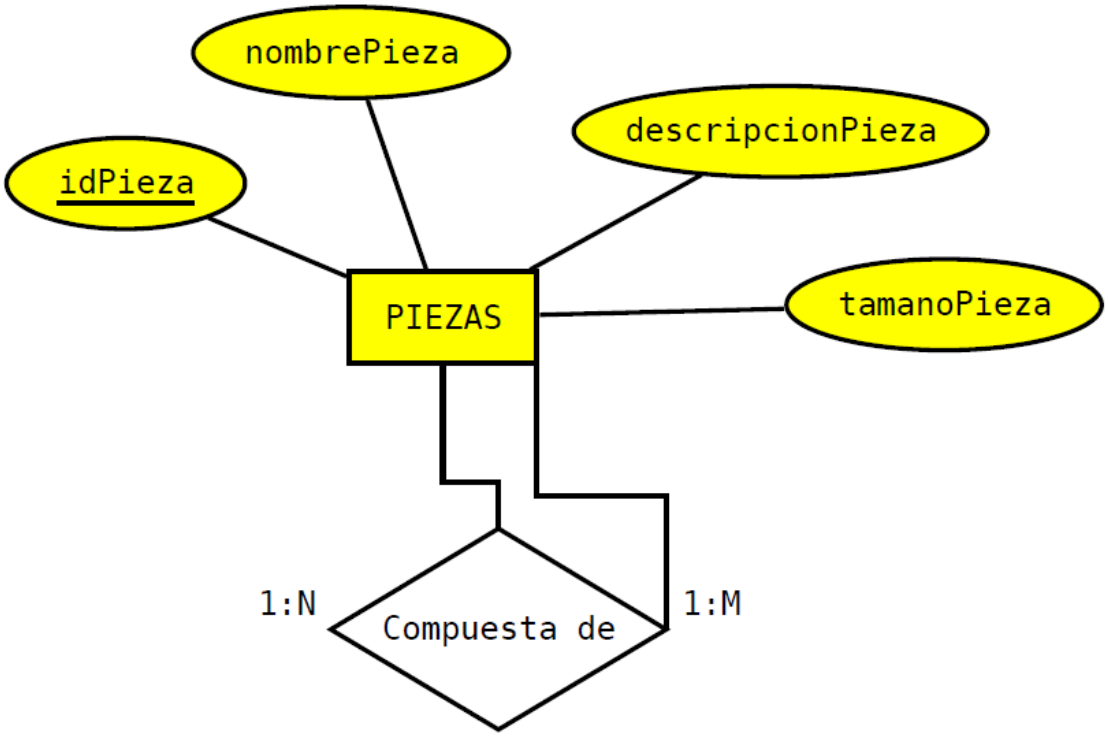
1.2. Si la relación es 1:N se procede como en el caso de relaciones 1:1.



El esquema relacional sería:

ALUMNOS (**idAlumno**, nombreAlumno, direccionAlumno, telefonoAlumno, idAlumnoDelegadoFK)

1.3. Si la relación es N:M, se trata igual que en las relaciones binarias. La tabla resultante de la relación contendrá 2 veces la clave primaria de la entidad del lado muchos, más los atributos de la relación si los hubiera. La clave de esta nueva tabla será la combinación de las 2.



El esquema relacional sería:

PIEZAS (**idPieza**, nombrePieza, descripcionPieza, tamanoPieza)
COMPONENTES (**idPiezaFK**, **idPiezaCompuestaFK**)

Por último, hay que comentar que cuando pasamos del **Diagrama Entidad-Relación** al **Esquema Relacional**, podemos empezar por cualquier Entidad para obtener su Esquema Relacional, pero una vez tengamos todas las tablas, es recomendable revisar el orden en que aparecen, pues a la hora de implementar esta base de datos, podemos cometer errores de creación.

Supongamos el siguiente Esquema Relacional:

ALUMNOS (**idAlumno**, nombreAlumno, direccionAlumno, telefonoAlumno, idCursoFK, fechaInicio)
CURSOS (**idCurso**, nombreCurso, turnoCurso)

Si nos disponemos a pasar este Esquema Relacional a una base de datos real en MySQL, por ejemplo, nos dará error la creación de Alumnos. El motivo es simple, en su creación, está haciendo referencia a la tabla Cursos con el campo idCursoFK. Y esta tabla aún no existe. Sería correcto creando primero Cursos y luego Alumnos. Como norma general, crear primero las tablas que no tienen campos FK. Y luego las demás tablas.

2. **Generalizaciones o especializaciones**: Las diferentes opciones para la transformación de jerarquías del modelo E-R al modelo relacional son:

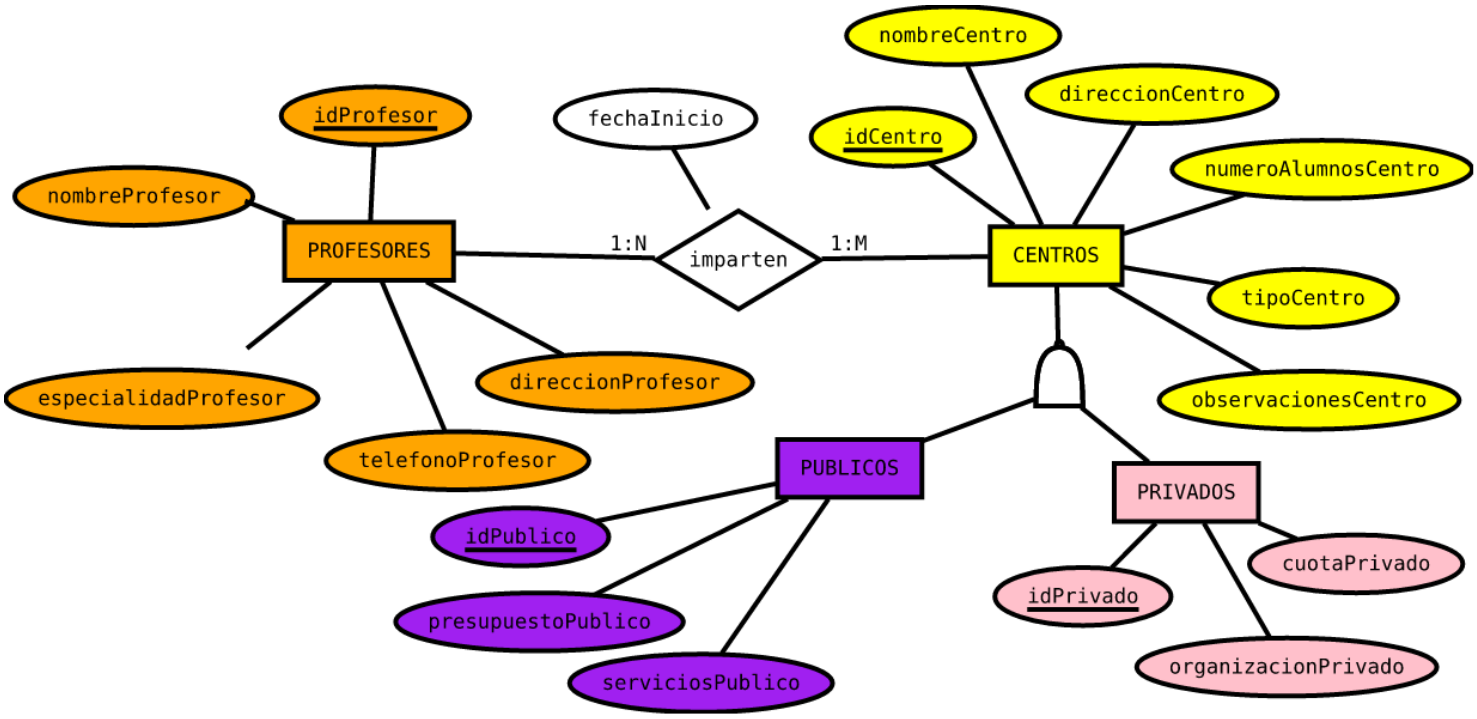
2.1. Integrar todas las entidades en una única tabla absorbiendo los subtipos: Se crea una tabla que contiene todos los atributos del supertipo, todos los de los subtipos, y el atributo discriminatorio para distinguir a qué subtipo pertenece cada registro de la tabla. Esta regla puede aplicarse a cualquier tipo de jerarquía, es muy simple de realizar, pero genera demasiados valores nulos en los atributos opcionales propios de cada subtipo.

2.2. Eliminación del supertipo en jerarquías totales y exclusivas: Transfiriendo los atributos del supertipo a cada uno de los subtipos, creándose una tabla por cada subtipo, el supertipo no tendrá tabla, y se elimina el atributo que distingue entre subtipos. Se crea redundancia en la información pues los atributos del supertipo se repiten en cada uno de los subtipos. El número de relaciones aumenta, pues las relaciones del supertipo pasan a cada uno de los subtipos.

2.3. Insertar una relación 1:1 entre el supertipo y los subtipos: Los atributos se mantienen y cada subtipo se identificará con una clave ajena referenciando a la clave primaria del supertipo. El supertipo mantendrá una relación 1:1 con cada subtipo.

Ejemplo de los Profesores y los centros especializados en públicos y privados:

- Hay profesores que imparten clases en 2 tipos de centros educativos: públicos y privados.
- Un profesor puede impartir clase en varios centros, ya sean públicos o privados.
- La asignatura será un atributo de la relación entre el profesor y el centro donde imparte.
- Los centros educativos sólo pueden ser públicos o privados.
- Un centro público no puede ser privado a la vez, ni a la inversa.
- Los atributos específicos para los centros públicos son: el presupuesto y los servicios; y para los privados son: la organización y la cuota.



Solución:

PROFESORES (**idProfesor**, nombreProfesor, direccionProfesor, telefonoProfesor, especialidadProfesor)
CENTROS (**idCentro**, nombreCentro, direccionCentro, numeroAlumnosCentro, tipoCentro)
PUBLICOS (**idPublico**, serviciosPublico, presupuestoPublico, idCentroFK)
PRIVADOS (**idPrivado**, organizacionPrivado, cuotaPrivado, idCentroFK)
IMPARTICIONES (**idProfesorFK**, **idCentroFK**, asignatura)

Otra posible solución resolviendo la generalización de otra forma:

PROFESORES (**idProfesor**, nombreProfesor, direccionProfesor, telefonoProfesor, especialidadProfesor)
CENTROS (**idCentro**, nombreCentro, direccionCentro, numeroAlumnosCentro, tipoCentro)
PUBLICOS (**idCentroFK**, serviciosPublico, presupuestoPublico)
PRIVADOS (**idCentroFK**, organizacionPrivado, cuotaPrivado)
IMPARTICIONES (**idProfesorFK**, **idCentroFK**, asignatura)

3. **Relaciones N-arias** (ternarias, cuaternarias, etc.): En este tipo de relaciones se asocian 3 o más entidades. Se pasan todas las entidades a tablas tal cual. La relación también se convierte a una tabla, que va a contener sus atributos más las claves primarias de todas las entidades que asocia como claves ajenas. Hay 2 casos:

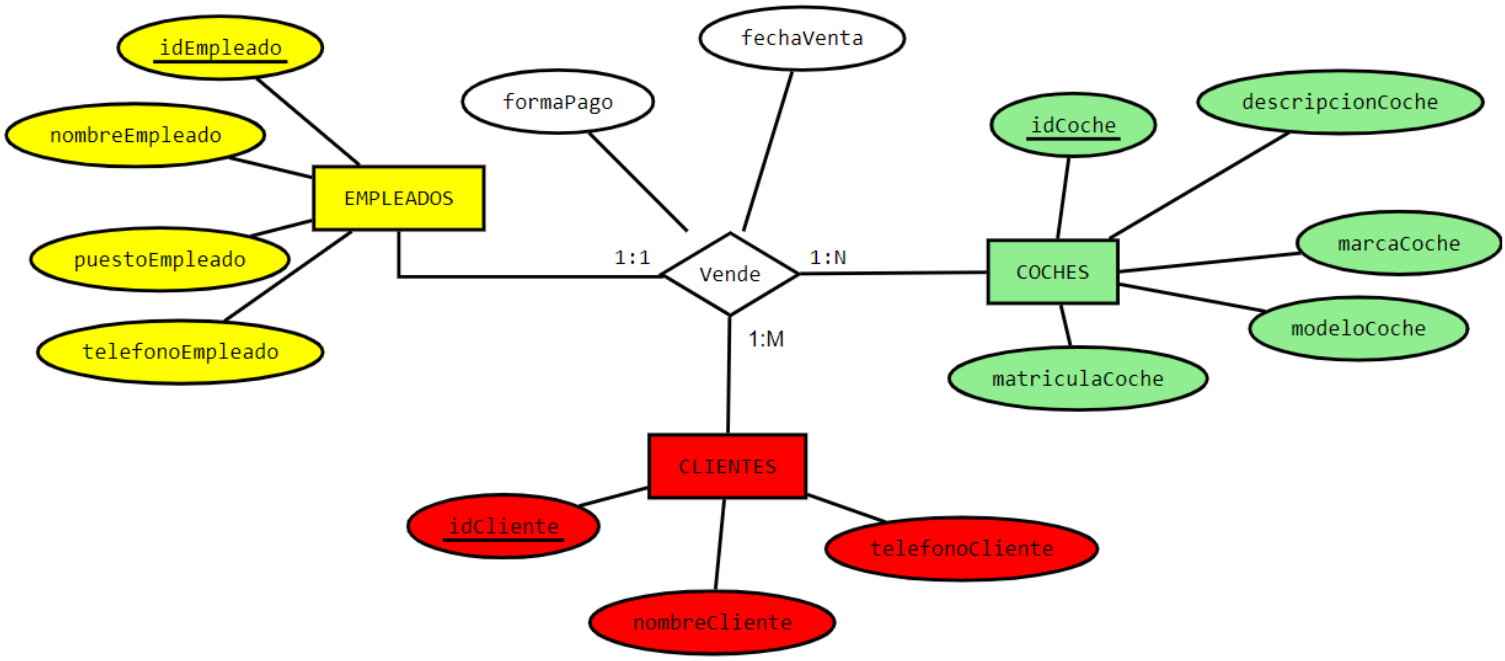
3.1. Si la relación es N:N:N, es decir, todas las entidades participan con cardinalidad máxima N, la clave de la tabla resultante de la relación es la unión de las claves ajenas que referencian a las entidades que asocia.

3.2. Si la relación es 1:N:N, es decir, una sola entidad participa con cardinalidad máxima 1, la clave de la tabla resultante de la relación es la

unión de las claves ajenas que referencian a las entidades que asocia excepto la de la entidad que participa con cardinalidad máxima 1, que queda como un atributo más y como clave ajena, pero no formará parte de la clave primaria de dicha tabla resultante de la relación.

Ejemplo: Vendedores de coches (empleados, clientes y coches):

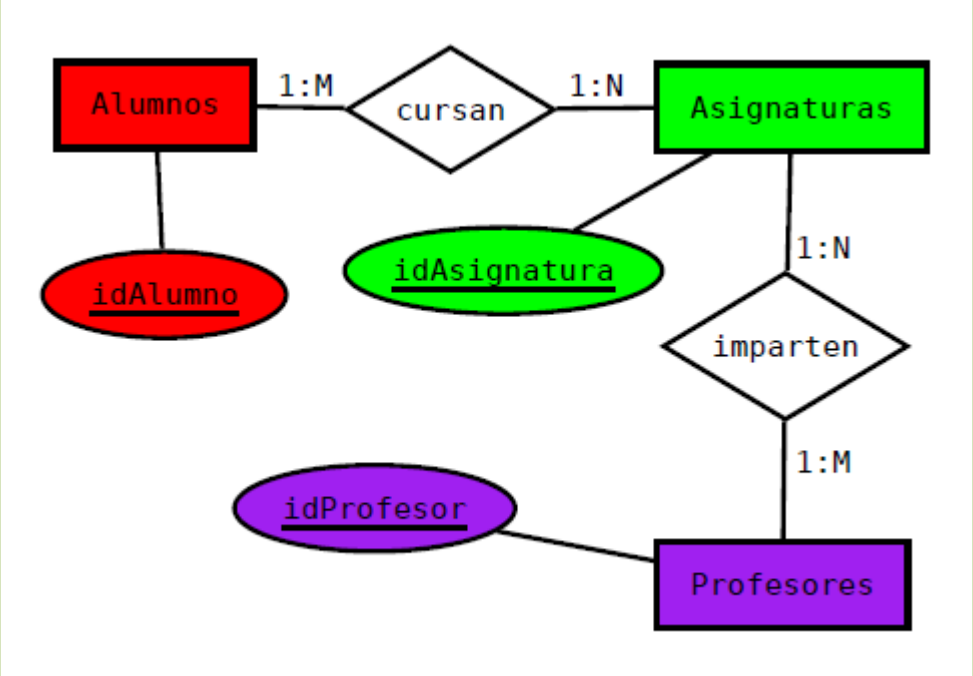
- En una tienda de coches, un empleado vende coches a sus clientes.
- En cada venta, un único empleado puede vender varios coches a varios clientes.
- En una operación de venta hay que tener en cuenta la forma de pago y la fecha de venta.



El resultado en el modelo relacional (siendo una relación ternaria del tipo 1:N:N) es:

EMPLEADOS (**idEmpleado**, nombreEmpleado, puestoEmpleado, telefonoEmpleado)
COCHES (**idCoche**, descripcionCoche, marcaCoche, modeloCoche, matriculaCoche)
CLIENTES (**idCliente**, nombreCliente, telefonoCliente)
VENTAS (**idEmpleadoFK**, **idCocheFK**, **idClienteFK**, formaPagoVenta, fechaVenta)

Ejemplo 5: Dado el siguiente Diagrama Entidad Relación (ERD) y su correspondiente Esquema Relacional (ER) transformarlo en una **relación ternaria** y obtener su correspondiente Esquema Relacional (ER).



Y su ER:

Alumnos (**idAlumno**, ...)

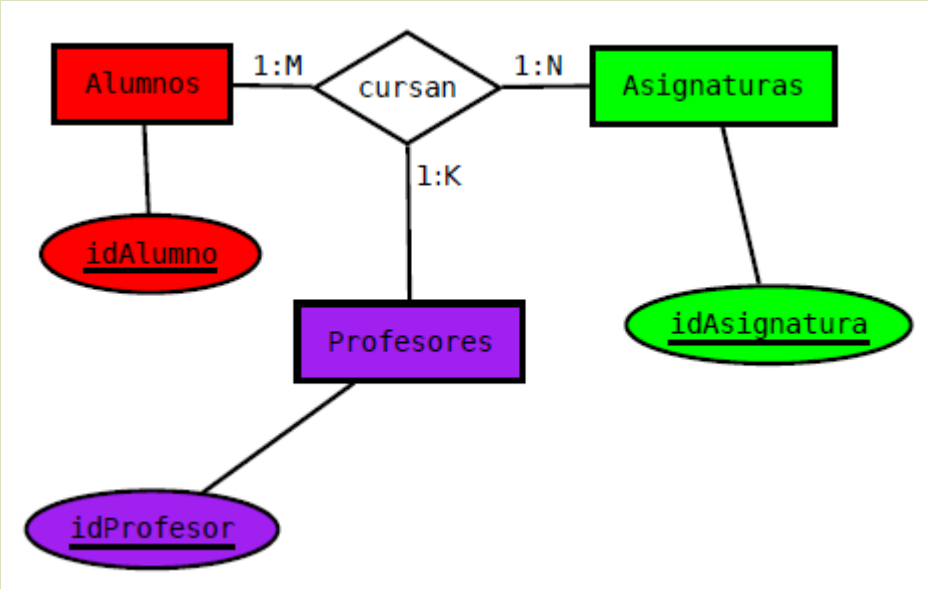
Asignaturas (**idAsignatura**, ...)

Profesores (**idProfesor**, ...)

Cursos (**idAlumnoFK**, **idAsignaturaFK**, ...)

Imparticiones (**idProfesorFK**, **idAsignaturaFK**, ...)

La solución sería (ERD):



Y su Esquema Relacional (ER):

Alumnos (**idAlumno**, ...)

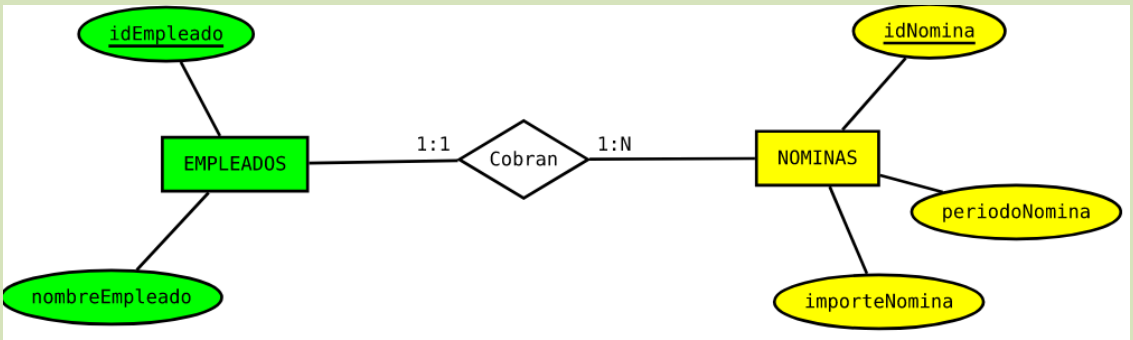
Asignaturas (**idAsignatura**, ...)

Profesores (**idProfesor**, ...)

Cursos (**idAlumnoFK**, **idAsignaturaFK**, **idProfesorFK**, ...)

Ejemplo 6: Obtener el Diagrama Entidad-Relación (ERD) y su correspondiente Esquema Relacional (ER) para recoger el supuesto de Empleados que cobran Nóminas en una empresa.

La solución sería (ERD):



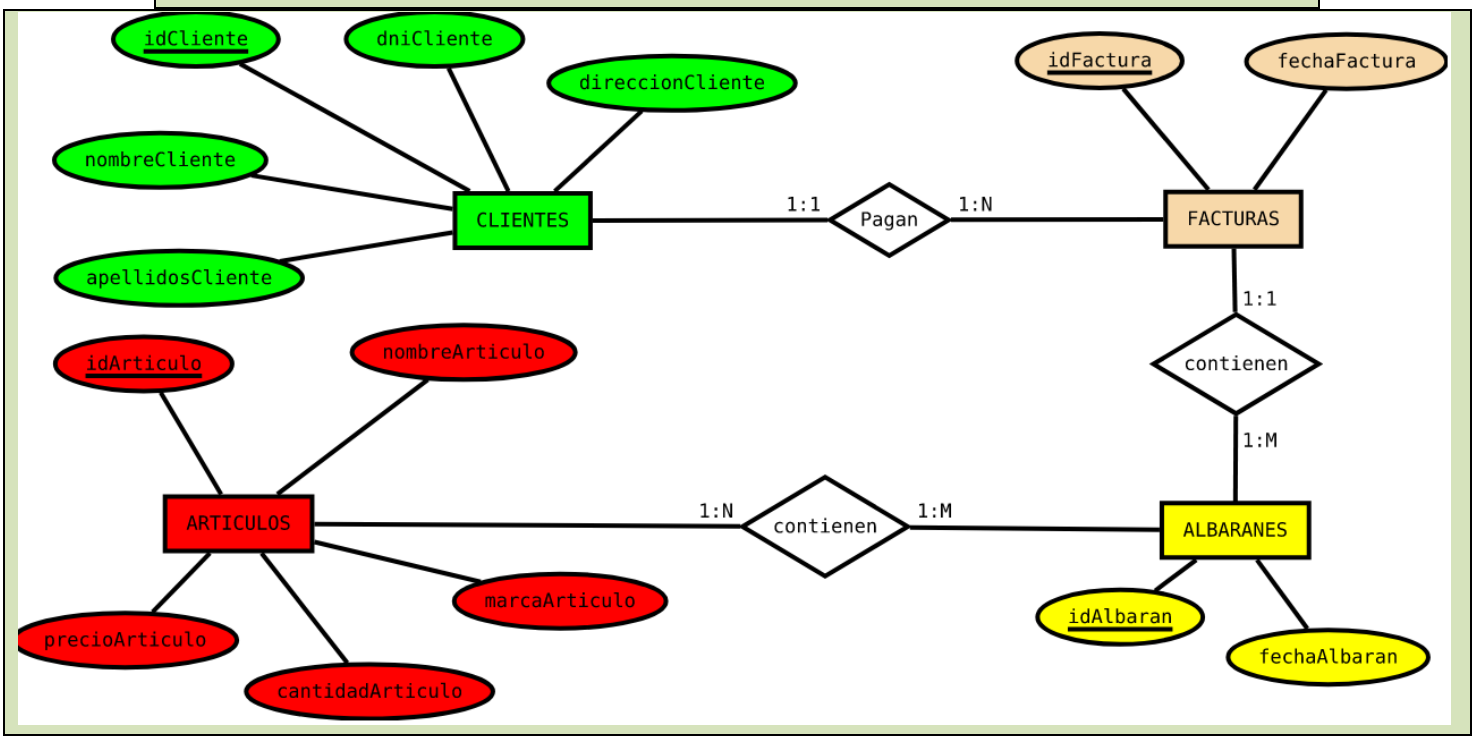
Y su Esquema Relacional (ER):

Empleados (**idEmpleado**, nombreEmpleado, ...)

Nominas (**idNomina**, periodoNomina, importeNomina, ..., idEmpleadoFK)

Ejemplo 7: Obtener el Diagrama Entidad-Relación (ERD) y su correspondiente Esquema Relacional (ER) para el supuesto de Clientes-Albaranes-Facturas-Artículos.

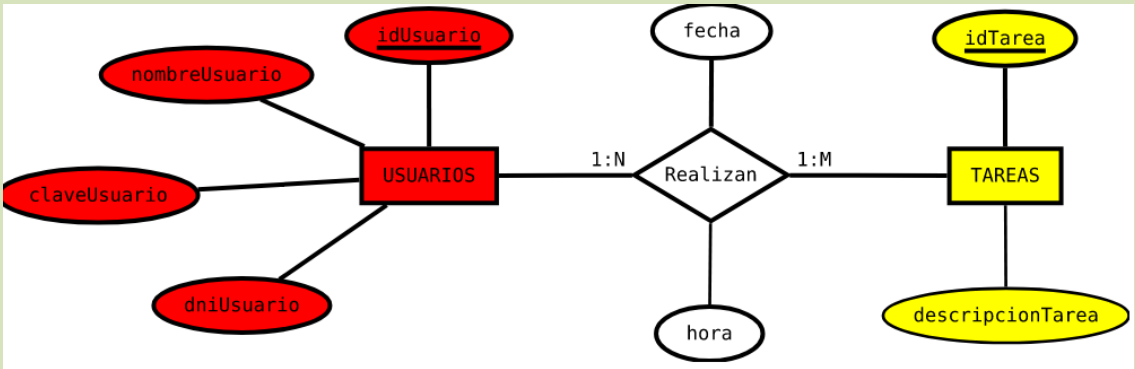
La solución sería (ERD):



Y el Esquema Relacional (ER):
Clientes (**idCliente**, dniCliente, nombreCliente, apellidosCliente, direccionCliente)
Facturas (**idFactura**, fechaFactura, idClienteFK)
Articulos (**idArticulo**, nombreArticulo, precioArticulo, cantidadArticulo, marcaArticulo)
Albaranes (**idAlbaran**, fechaAlbaran, idFacturaFK)
LineasAlbaran (**idLineaAlbaran**, idArticuloFK, idAlbaranFK)
LineasAlbaran (**idArticuloFK**, **idAlbaranFK**)

Ejemplo 8: Obtener el Diagrama Entidad-Relación (ERD) y su correspondiente Esquema Relacional (ER) para el supuesto de un histórico de las tareas realizadas en un sistema por los usuarios de este.

La solución sería (ERD):



Y el Esquema Relacional (ER):

Usuarios (**idUsuario**, nombreUsuario, claveUsuario, dniUsuario)

Tareas (**idTarea**, descripcionTarea)

Historico (**idHistorico**, idUsuarioFK, idTareaFK, fecha, hora)

Historico (**idUsuarioFK**, **idTareaFK**, fecha, hora)

24/08/2022