INTRODUCCION

Lo primero y fundamental para ingresar a la programación con Bases de Datos, es lo siguiente:

Etapas de un proyecto.

En la figura vemos bien definidas las tres etapas de todo proyecto. La primera es **DATOS**, que son los elementos básicos o fundamentales con los que cuento en el momento de inicio de la tarea a realizar, por si solos, ellos no poseen ningún valor agregado. La segunda es **PROCESO**, que es un conjunto de acciones o tareas que se realizan en un orden específico con un principio y un fin bien definidos. Por último, la tercera es **INFORMACION** y es el resultado que se obtiene como consecuencia de procesar los datos, y el fin primordial que se busca en todo sistema o programa informático es la obtención de información certera y precisa en el menor tiempo posible.

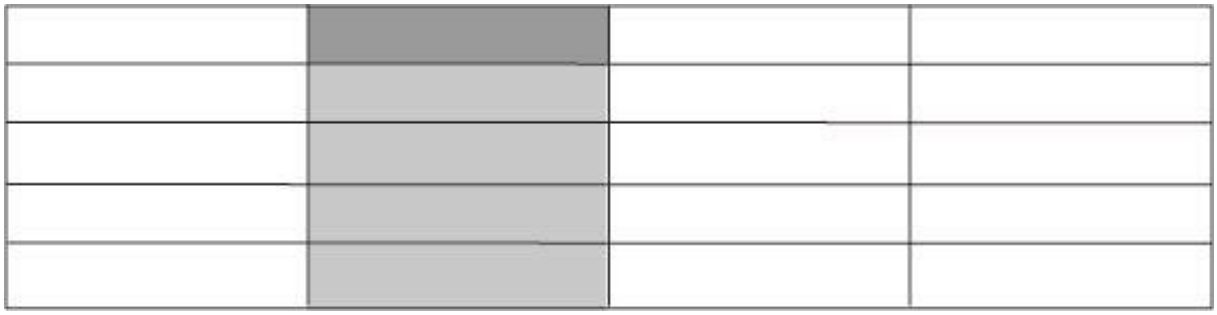
Bases de Datos Relacionales

Una Base de Datos relacional consiste en un conjunto de tablas o relaciones, donde cada una de ellas varía o puede variar con el transcurso del tiempo y se identifica de manera única por medio de un nombre.

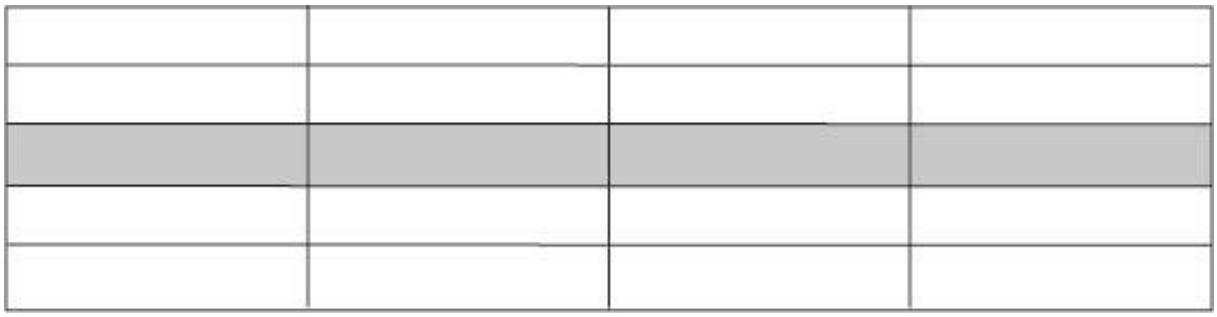
¿Qué es una tabla?

Una tabla es el único objeto de la base de datos en la que se guardan los datos. Está conformada por campos y registros formando una celda en la intersección de estos.

Campo o Atributo: También llamado Propiedad son los elementos que componen las entidades. Representa por ejemplo, los apellidos de todos nuestros clientes, el teléfono, etc. Se representa por columna.



Registro: Equivale a una fila de atributos en el cual consta todas las propiedades de un mismo miembro de datos. Por ejemplo, los datos personales de un Alumno (Nombre, Dirección, Teléfono, CP, email, etc.).



Tipos de Datos

Dentro de la definición de las tablas nos encontraremos con distintos tipos de datos que podemos asignarle a cada campo, estos tipos de datos difieren según la base de datos con la cual nos encontremos trabajando, datos numéricos (enteros chicos, grandes, decimales, etc.), fecha y hora, cadenas de caracteres, booleanos, etc.

En el sentido más amplio, se podría considerar que una base de datos es simplemente un conjunto de información (básicamente una serie de "hojas con una determinada estructura").
Por ejemplo, una base de datos muy sencilla podría ser una agenda de direcciones en la que anotemos datos de nuestros amigos. Tendríamos todas las letras con espacios para ingresar a cada uno de estos amigos. En cada espacio, a su vez, existirá una serie de apartados, como el nombre, la dirección, el teléfono, etc.

Nombre	*Dirección*	*Ciudad*	*Teléfono*
José Luis	Ruta 197 2312	Pacheco	1111-1111
Mariana	Av. Cazón 1239	Tigre	2222-2222
Juan	Estrada 231	Pacheco	3333-3333

Pero en la práctica, una "**base de datos**" real suele estar formada por más de una tabla. Por ejemplo, la base de datos que utiliza una empresa "normal" para su gestión deberá almacenar datos sobre clientes, proveedores, artículos, facturas, etc.
Cada uno de estos "bloques" de datos será una tabla, y estas tablas estarán relacionadas entre sí (por ejemplo: un artículo será suministrado por un cierto proveedor, y ese artículo aparecerá en ciertas facturas, cada una de las cuales corresponderá a un cierto cliente).
Todo este conjunto de información que forman las tablas, las relaciones entre ellas y algunas cosas más será nuestra "base de datos".

- *Base de Datos es un **conjunto exhaustivo no redundante** de datos **estructurados organizados independientemente de su utilización** y su implementación en máquina **accesibles en tiempo real** y compatibles con u**suarios concurrentes con necesidad de información diferente**.*

Nosotros trabajaremos en concreto, sobre lo que se conoce como una "**base de datos relacional**".

Tipos de bases de datos

Las bases de datos pueden clasificarse de varias maneras, de acuerdo al criterio elegido para su clasificación:

Según la variabilidad de los datos almacenados

Bases de datos estáticas

Éstas son bases de datos de sólo lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones y tomar decisiones.

Bases de datos dinámicas

Éstas son bases de datos donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización y adición de datos, además de las operaciones fundamentales de consulta. Un ejemplo de esto puede ser la base de datos utilizada en un sistema de información de un negocio determinado, una farmacia, una clínica, etc.

Según el contenido

Bases de datos bibliográficas

Solo contienen un representante de la fuente primaria, que permite localizarla. Un registro típico de una base de datos bibliográfica contiene información sobre el autor, fecha de publicación, editorial, título, edición, de una determinada publicación, etc. Puede contener un resumen o extracto de la publicación original, pero nunca el texto completo, porque sino estaríamos en presencia de una base de datos a texto completo (o de fuentes primarias).

Bases de datos de texto completo

Almacenan las fuentes primarias, como por ejemplo, todo el contenido de todas las ediciones de una colección de revistas científicas.

Directorios

Un ejemplo son las guías telefónicas en formato electrónico.

Bases de datos o "bibliotecas" de información Biológica

Son bases de datos que almacenan diferentes tipos de información proveniente de las ciencias de la vida o médicas, o bien como su nombre lo indica, el contenido son cifras o números. Por ejemplo, una colección de resultados de análisis de laboratorio, entre otras. Se pueden considerar en varios subtipos:

- Aquellas que almacenan secuencias de nucleótidos o proteínas.
- Las bases de datos de rutas metabólicas
- Bases de datos de estructura, comprende los registros de datos experimentales sobre estructuras 3D de biomoléculas
- Bases de datos clínicas
- Bases de datos bibliográficas (biológicas)

Modelos de bases de datos

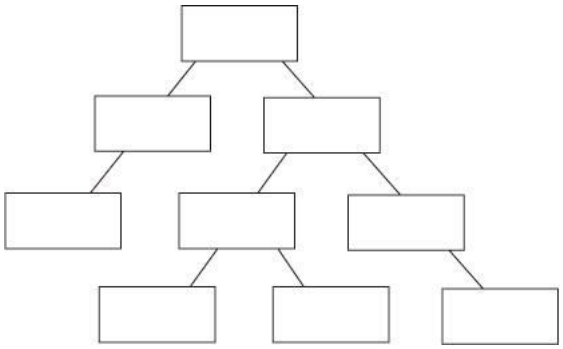
Además de la clasificación por la función de las bases de datos, éstas también se pueden clasificar de acuerdo a su modelo de administración de datos.

Un modelo de datos es básicamente una "descripción" de algo conocido como *contenedor de datos* (algo en donde se guarda la información), así como de los métodos para almacenar y recuperar información de esos contenedores. Los modelos de datos no son cosas físicas: son abstracciones que permiten la implementación de un sistema eficiente de *base de datos*; por lo general se refieren a algoritmos, y conceptos matemáticos. Un algoritmo es una lista bien definida, ordenada y finita de operaciones que permite hallar la solución a un problema.

Algunos modelos con frecuencia utilizados en las bases de datos:

Bases de datos jerárquicas

Éstas son bases de datos que, como su nombre indica, almacenan su información en una estructura jerárquica. En este modelo los datos se organizan en una forma similar a un árbol (visto al revés), en donde un *nodo padre* de información puede tener varios *hijos*. El nodo que no tiene padres es llamado *raíz*, y a los nodos que no tienen hijos se los conoce como *hojas*.

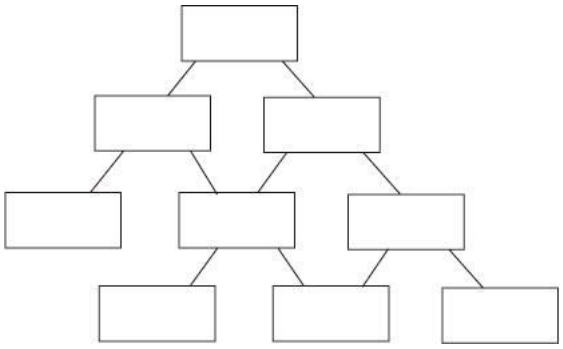


Las bases de datos jerárquicas son especialmente útiles en el caso de aplicaciones que manejan un gran volumen de información y datos muy compartidos permitiendo crear estructuras estables y de gran rendimiento.

Una de las principales limitaciones de este modelo es su incapacidad de representar eficientemente la redundancia de datos.

Base de datos de red

Éste es un modelo ligeramente distinto del jerárquico; su diferencia fundamental es la modificación del concepto de *nodo*: se permite que un mismo nodo tenga varios padres (posibilidad no permitida en el modelo jerárquico).

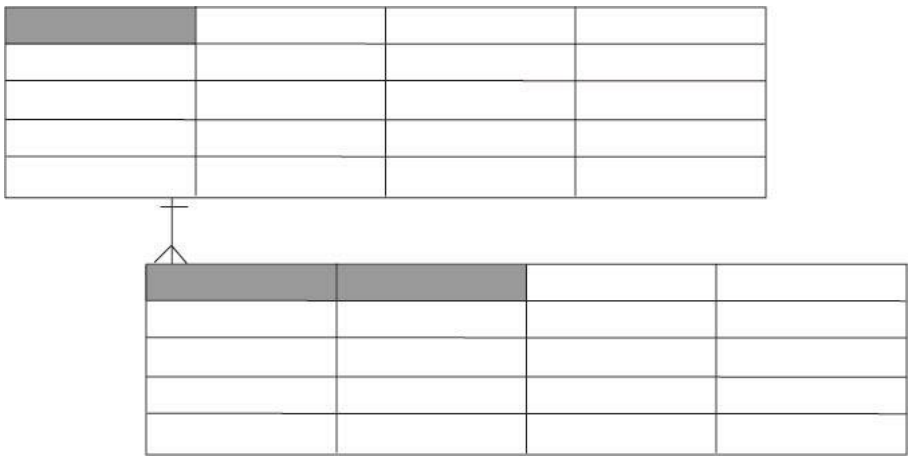


Fue una gran mejora con respecto al modelo jerárquico, ya que ofrecía una solución eficiente al problema de redundancia de datos; pero, aun así, la dificultad que significa administrar la información en una base de datos de red ha significado que fuera un modelo utilizado en su mayoría por programadores más que por usuarios finales.

Base de datos relacional

Éste es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Tras ser postulados sus fundamentos en 1970 por Edgar Frank Codd, de los laboratorios IBM en San José (California), no tardó en consolidarse como un nuevo paradigma en los modelos de base de datos. Su idea fundamental es el uso de "relaciones". Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados "tuplas". Una **tupla**, en matemáticas, es una secuencia ordenada de objetos, esto es, una lista con un número limitado de objetos (una secuencia *infinita* se denomina en matemática como una familia). Pese a que ésta es la teoría de las bases de datos relacionales creadas por Edgar Frank Codd, la mayoría de las veces se conceptualiza de una manera más fácil de imaginar. Esto es pensando en cada relación como si fuese una tabla que está compuesta por *registros* (las filas de una tabla), que representarían las tuplas, y *campos* (las columnas de una tabla).

En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia (a diferencia de otros modelos como el jerárquico y el de red). Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario esporádico de la base de datos. La información puede ser recuperada o almacenada mediante "consultas" que ofrecen una amplia flexibilidad y poder para administrar la información.



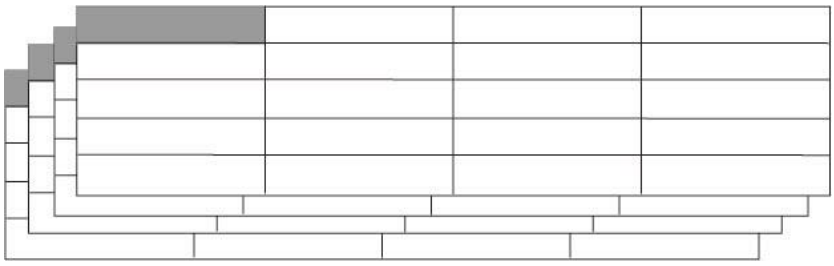
El lenguaje más habitual para construir las consultas a bases de datos relacionales es SQL, *Structured Query Language* o *Lenguaje Estructurado de Consultas*, un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales.

Durante su diseño, una base de datos relacional pasa por un proceso al que se le conoce como normalización de una base de datos.

Durante los años '80 (1980-1989) la aparición de dBASE produjo una revolución en los lenguajes de programación y sistemas de administración de datos. Aunque nunca debe olvidarse que dBase no utilizaba SQL como lenguaje base para su gestión.

Bases de datos multidimensionales

Son bases de datos ideadas para desarrollar aplicaciones muy concretas, como creación de Cubos OLAP (Procesamiento Analítico On-Line). Básicamente no se diferencian demasiado de las bases de datos relacionales (una tabla en una base de datos relacional podría serlo también en una base de datos multidimensional), la diferencia está más bien a nivel conceptual; en las bases de datos multidimensionales los campos o atributos de una tabla pueden ser de dos tipos, o bien representan dimensiones de la tabla, o bien representan métricas que se desean estudiar.



Bases de datos orientadas a objetos

Este modelo, el mas reciente, y propio de los modelos informáticos orientados a objetos, trata de almacenar en la base de datos los *objetos* completos (estado y comportamiento).

Una base de datos orientada a objetos es una base de datos que incorpora todos los conceptos importantes del paradigma de objetos:

- Encapsulación - Propiedad que permite ocultar la información al resto de los objetos, impidiendo así accesos incorrectos o conflictos.
- Herencia - Propiedad a través de la cual los objetos heredan comportamiento dentro de una jerarquía de clases.
- Polimorfismo - Propiedad de una operación mediante la cual puede ser aplicada a distintos tipos de objetos.

En bases de datos orientadas a objetos, los usuarios pueden definir operaciones sobre los datos como parte de la definición de la base de datos. Una operación (llamada función) se especifica en dos partes. La interfaz (o signatura) de una operación incluye el nombre de la operación

y los tipos de datos de sus argumentos (o parámetros). La implementación (o método) de la operación se especifica separadamente y puede modificarse sin afectar la interfaz. Los programas de aplicación de los usuarios pueden operar sobre los datos invocando a dichas operaciones a través de sus nombres y argumentos, sea cual sea la forma en la que se han implementado. Esto podría denominarse independencia entre programas y operaciones.

Bases de datos documentales

Permiten la indexación a texto completo, y en líneas generales realizar búsquedas más potentes. Tesaurus es un sistema de índices optimizado para este tipo de bases de datos.

Base de datos deductivas

Un sistema de **base de datos deductivas**, es un sistema de base de datos pero con la diferencia de que permite hacer deducciones a través de inferencias. Se basa principalmente en reglas y hechos que son almacenados en la base de datos. También las bases de datos deductivas son llamadas **base de datos lógica**, a raíz de que se basan en lógica matemática.

Bases de datos distribuidas

La base de datos está almacenada en varias computadoras conectadas en red. Surgen debido a la existencia física de organismos descentralizados. Esto les da la capacidad de unir las bases de datos de cada localidad y acceder así a distintas universidades, sucursales de tiendas, etc.

Bases de datos NoSQL

Cuando hablamos de bases de datos NoSQL nos referimos a una amplia clase de sistemas de gestión de datos (mecanismos para el almacenamiento y recuperación de datos) que difieren, en aspectos importantes, del modelo clásico de relaciones entre entidades (o tablas) existente en los sistemas de gestión bases de datos relacionales, siendo el más destacado el que no usan SQL como lenguaje principal de consulta.

Pese a todas las opciones proporcionadas por el auge de las bases de datos NoSQL, esto no significa la desaparición de las bases de datos de RDBMS ya que son tecnologías complementarias. Estamos entrando en una era de persistencia políglota, una técnica que utiliza diferentes tecnologías de almacenamiento de datos para manejar las diversas necesidades de almacenamiento de datos.

https://blogs.oracle.com/spain/qu-es-una-base-de-datos-nosql **(consultado el 11 de marzo de 2019)**

Etapas básicas del diseño.

Primero y principal, no te apresures a escribir código nunca. Si no diseñas primero la solución, te vas a encontrar después con miles de problemas en el camino en los que vas a gastar más tiempo del que hubieras gastado realizando un buen diseño.

Lo primero será documentar bien los requerimientos, la necesidad que vas a resolver/solucionar, los datos, como agruparlos y sus restricciones. Con los requerimientos bien documentados podrás ir diseñando cada uno de los objetos de tu base de datos. En nuestra cursada utilizaremos a forma de relevamiento los enunciados de los ejercicios.

Generalmente, los requerimientos bien documentados en un buen relevamiento son todo lo que necesitas para comenzar a diseñar tus bases de datos.

- **Diseño conceptual** - D.E.R. (Diagrama Entidad Relación)

Se trata de un modelo para representar nuestras tablas (entidades) y las relaciones que existirán entre ellas. En esta etapa se recomienda realizarlo a mano, en una hoja donde podremos tachar, borronear y volver a escribir, o en una aplicación para representar este tipo de diagramas, teniendo en cuenta que se trata de un diagrama previo a la creación de la base de datos.

Nunca confundir un DER con un Diagrama de Base de Datos

- **Diseño lógico**

En esta fase, debemos pensar en los campos que conformarán cada una de nuestras tablas y en cómo normalizar nuestras tablas para evitar duplicidad de información y para ahorrar espacio de almacenamiento. Esto último (ahorrar espacio) ya no es tan importante como hace algunos años, pero es necesario para optimizar el funcionamiento de una base de datos relacional

- **Diseño físico**

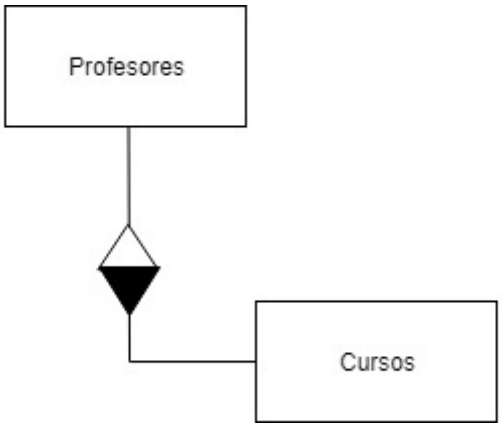
En esta última fase ya debemos revisar con detalle los tipos de datos que utilizaremos, sus dominios (qué valores va a permitir), cuales índices debemos crear para optimizar las consultas, entre otros. Aquí ya transformaremos nuestro diseño en una base de datos física, ya sea mediante la interface gráfica o escribiendo nuestro SQL en el motor de bases de datos elegido.

Introducción al modelo Entidad-Relación.

Este es un modelo que nos permitirá "dibujar" las entidades y las relaciones que existen entre ellas. Nosotros usaremos un "Diagrama Entidad-Relación" (**DER**, de aquí en adelante). Existen varias notaciones ligeramente distintas. Vamos a utilizar la que consideramos más sencilla.
En esta notación se representan las entidades como un rectángulo y las relaciones binarias como un rombo partido por la mitad. Las relaciones siempre serán para nosotros de uno a muchos (1:M), una de las mitades (la que corresponde al "muchos") deberá estar sombreada.

(Nota: hemos visto las relaciones expresadas también de otras formas, por otros autores; de momento, emplearemos esta notación y más adelante comentaré otras notaciones que es posible utilizar o encontrar en otros textos).

Diagrama ER de un ejemplo.



Así de sencillo: tenemos 2 entidades (Profesores y Cursos) y una relación (impartir, entre profesores y cursos, 1:M).

Pensemos primero qué campos o atributos nos podrían interesar para integrar nuestras tablas o entidades, agregando también una tabla de "alumnos".

Profesores:
- DNI.
- Nombre
- Dirección
- Ciudad
- Teléfono
- Título
- Sueldo
- Cuenta bancaria

Cursos:
- Nombre del curso
- Fecha de comienzo
- Duración (horas)
- Importe (pesos)
- Número máximo de alumnos

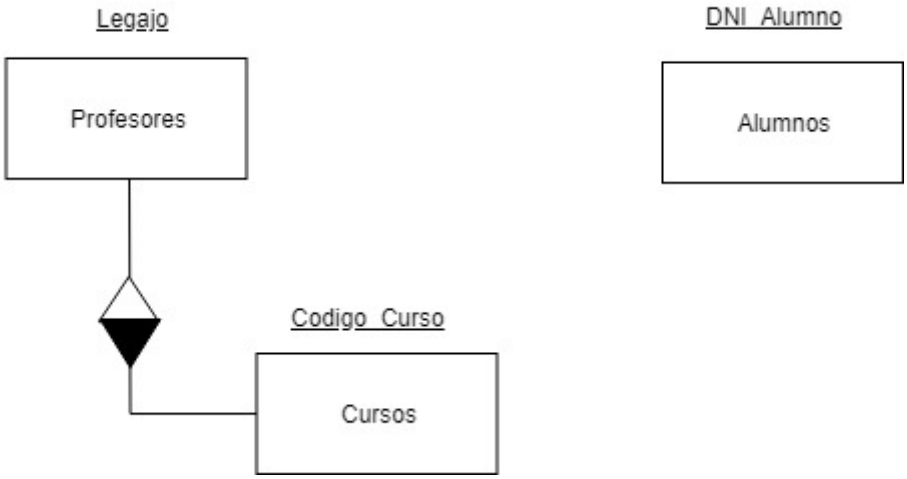
Alumnos:
- Legajo
- DNI (Documento Nacional de Identidad)
- Nombre
- Dirección
- Ciudad
- Teléfono
- Fecha de nacimiento
- Fecha de alta en el centro
- Fotografía

Es sólo un ejemplo. En este momento estamos pensando en qué datos queremos almacenar, lo que debería ser el resultado de los requerimientos de nuestro relevamiento.
Deberíamos pensar en qué tipo de dato le asignaremos a cada campo de cada tabla, teniendo en cuenta que los únicos campos a los que les asignaremos tipo de dato numérico son aquellos que serán calculables, y que según el sistema de bases de datos que empleemos realmente, puede ocurrir que sea incómodo (o incluso imposible) trabajar con algunos de estos datos que hemos previsto (por ejemplo, la "fotografía" del alumno).
Luego definiremos cual de esos datos nos permitirá **distinguir un registro de otro**. Esto se hace porque podemos tener dos alumnos con el mismo nombre, pero claramente son personas distintas, y debemos saber qué cursos ha realizado cada uno de ellos sin posibilidad de confusión, para no dar a uno el diploma que corresponda a otro, ni cobrarle un dinero de otro.
En el caso de los alumnos, no son datos únicos los siguientes: el nombre (puede repetirse, incluso con apellidos), la dirección (dos hermanos o dos amigos pueden vivir en la misma casa), el teléfono (ocurre lo mismo), la fecha de nacimiento (también podemos encontrar dos alumnos que hayan nacido el mismo día), etc. Lo que realmente distinguirá a un alumno de otro es su número

de DNI (Documento Nacional de Identidad), o pasaporte, o CUIT/CUIL, que sí es único.
Pues bien, este dato que puede distinguir un alumno de otro (o en general un registro de otro) es lo que llamaremos la "**clave"**.
Puede ocurrir que no exista nada que nos sirva claramente como clave, como es el caso de los cursos: no es único el nombre (podemos impartir más de un curso con el mismo contenido), ni la fecha de comienzo (varios cursos pueden comenzar el mismo día), ni la duración, ni el importe, ni el número máximo de alumnos. En estos casos se suele añadir algo arbitrario, un **código**, que nos permita distinguir un curso de otro (en general un registro de otro). En nuestro caso, incluiríamos un nuevo atributo, llamado "Código de curso".

Debemos tener en cuenta que para relacionar dos tablas debe existir entre ellas al menos un campo coincidente, con exactamente el mismo tipo de dato relacionando registros en base a una igualdad en sus campos coincidentes.

Un último comentario antes de ver cómo quedaría nuestro DER. Puede ocurrir que nuestra tabla tenga varios atributos únicos, todos los cuales puedan servir como clave. Entonces escogemos una de ellas (o varias) como "**clave principal**", (luego veremos más tipos de claves) y el resto serán "**datos**" que no llegaremos a usar como claves. En el diagrama, el atributo que vaya a utilizarse como clave principal aparecerá subrayado.
En este caso se muestra la representación de un DER (diagrama de entidad relación).
Ahora ya sí, nuestro diagrama quedaría así (no incluimos todos los atributos que habíamos pensado, sólo las claves, que es con los que trabajaremos a partir de ahora):



Teniendo en cuenta que nosotros dijimos que nuestras relaciones **siempre** van a ser de **uno a muchos** y que un alumno va a asistir a muchos cursos, pero un curso va a tener a muchos alumnos, necesitamos formar de alguna manera una relación de **muchos a muchos** para lo cual agregaremos una tabla auxiliar que va a relacionar a las dos que queremos relacionar de esta forma.