Clase 12

Bases de datos

12.1 Introducción

Una de las maneras de manejar la gran cantidad de datos e información de la que se dispone hoy en día es mediante las bases de datos. En la actualidad, las bases de datos se usan tan ampliamente que se pueden encontrar en organizaciones de todos los tamaños, desde grandes corporaciones y agencias gubernamentales, hasta pequeños negocios e incluso en hogares. Nuestras actividades diarias con frecuencia nos ponen en contacto con las bases de datos, ya sea directa o indirectamente. Las bases de datos nos permiten trabajar con grandes cantidades de datos facilitando las operaciones de alta, baja, consulta y modificación, entre otras.

Algunos ejemplos de aplicación de base de datos pueden ser:

Cuando visitamos un portal de compras online, que permite navegar y comprar artículos, se accede a una base de datos. La información acerca de los productos disponibles y los datos acerca del pedido se almacenan en una base de datos. También es posible que pueda ver los datos almacenados acerca de pedidos anteriores que haya realizado. Algunos sitios web pueden usar información acerca de sus pedidos, o incluso sus actividades de navegación, para sugerir productos o servicios que es probable que le interesen.

Otro caso es el uso de la banca electrónica, que permite recuperar registros de base de datos acerca de depósitos, retiros, pago de facturas y otras transacciones para sus cuentas. Además, podemos transferir fondos, ordenar cheques y realizar muchas otras funciones, todas las cuales involucran el uso de una base de datos.

Los registros académicos también se conservan en una base de datos que se actualiza cada periodo al registrar su inscripción, conclusión y calificación para cada materia.

Estos son algunos ejemplos de uso de las bases de datos se usan para satisfacer las necesidades de información de muchas organizaciones e individuos en una variedad de áreas.

En esta clase realizaremos una presentación de los fundamentos teóricos, estructura interna y diseño de las bases de datos.

12.2 Base de datos

En la evolución de la informática, a medida que se complejizaban e integraban las aplicaciones, se tuvo que interrelacionar su información y fue necesario eliminar la redundancia. El nuevo conjunto de información o datos se debía diseñar de modo que estuviesen interrelacionados; y, al mismo tiempo, las informaciones redundantes (como, por ejemplo, el nombre y la dirección de los clientes o el nombre y el precio de los productos), que figuraban en los ficheros de más de una de las aplicaciones, debían estar ahora en un solo lugar.

Estos conjuntos de ficheros interrelacionados, con estructuras complejas y compartidos por varios procesos de forma simultánea conforman lo que denominamos bases de datos (BD).

Una base de datos es un conjunto de datos almacenados en memoria externa que están organizados mediante una estructura de datos. Cada base de datos ha sido diseñada para satisfacer los requisitos de información de una empresa u otro tipo de organización, como por ejemplo, una universidad o un hospital.

Definición

Una **base de datos** es una recopilación organizada de información o datos estructurados, que normalmente se almacena de forma electrónica en un sistema informático. Normalmente, una base de datos está controlada por un sistema de gestión de bases de datos (SGBD, o DBMS en inglés). En conjunto, los datos y el SGBD, junto con las aplicaciones asociadas a ellos, reciben el nombre de sistema de bases de datos, comúnmente abreviado a **base de datos**.

Las bases de datos son ampliamente usadas. Las siguientes son algunas de sus aplicaciones más representativas:

- En bancos. Para información de los clientes, cuentas y préstamos, y transacciones bancarias.
- En aerolíneas. Para reservas e información de planificación. Las líneas aéreas fueron de los primeros en usar las bases de datos de forma distribuida geográficamente (los terminales situados en todo el mundo accedían al sistema de bases de datos centralizado a través de las líneas telefónicas y otras redes de datos).
- Universidades. Para información de los estudiantes, matrículas de las asignaturas y cursos.
- Transacciones con tarjetas de crédito. Para compras con tarjeta de crédito o débito y generación mensual de extractos.
- Telecomunicaciones. Para guardar un registro de las llamadas realizadas, ancho de banda consumido, generación mensual de facturas, etc.
- Finanzas. Para almacenar información sobre grandes empresas, ventas y compras de documentos formales financieros, como bolsa y bonos.
- Ventas. Para información de clientes, productos y compras.
- Producción. Para la gestión de la cadena de producción y para el seguimiento de la producción de elementos en las fábricas, inventarios de elementos en almacenes y pedidos de elementos.
- Recursos humanos. Para información sobre los empleados, salarios, impuestos y beneficios, etc.
- La tecnología Blockchain, sobre la que se basa el funcionamiento de criptomonedas como el Bitcoin, podría considerarse como la única base de datos que está totalmente descentralizada y no depende de ningún organismo.

12.2.1 Sistema gestor de base de datos

Un sistema gestor de bases de datos consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a dichos datos. La

colección de datos es normalmente denominada base de datos. El objetivo principal de un SGBD es proporcionar una forma de almacenar y recuperar la información de una base de datos de manera que sea tanto *práctica* como *eficiente*.

Los sistemas de bases de datos se diseñan para gestionar grandes cantidades de información.

Algunos de los sistemas de gestión de bases de datos más utilizados son:

- MySQL
- Microsoft SQL Server
- PostgreSQL
- SQLite
- Microsoft Access
- Oracle

La gestión de los datos implica tanto la definición de estructuras para almacenar la información como la provisión de mecanismos para la manipulación de la información. Además, los sistemas de bases de datos deben proporcionar la fiabilidad de la información almacenada, a pesar de las caídas del sistema o los intentos de acceso sin autorización. Si los datos van a ser compartidos entre diversos usuarios, el sistema debe evitar posibles resultados anómalos.

Generalmente, un SGBD proporciona los servicios que se citan a continuación:

- El SGBD permite la definición de la base de datos mediante un lenguaje de definición de datos. Este lenguaje permite especificar la estructura y el tipo de los datos, así como las restricciones sobre los datos.
- El SGBD permite la inserción, actualización, eliminación y consulta de datos mediante un lenguaje de manejo de datos. El hecho de disponer de un lenguaje para realizar consultas reduce el problema de los sistemas de ficheros, en los que el usuario tiene que trabajar con un conjunto fijo de

consultas, o bien, dispone de un gran número de programas de aplicación costosos de gestionar.

- Un sistema de seguridad, de modo que los usuarios no autorizados no puedan acceder a la base de datos.
- Un sistema de integridad que mantiene la integridad y la consistencia de los datos.
- Un sistema de control de concurrencia que permite el acceso compartido a la base de datos.
- Un sistema de control de recuperación que restablece la base de datos después de que se produzca un fallo del hardware o del software.
- Un diccionario de datos o catálogo, accesible por el usuario, que contiene la descripción de los datos de la base de datos.

Con estas funcionalidades, el SGBD se convierte en una herramienta de gran utilidad. Sin embargo, desde el punto de vista del usuario, se podría discutir que los SGBD han hecho las cosas más complicadas, ya que ahora los usuarios ven más datos de los que realmente quieren o necesitan, puesto que ven la base de datos completa. Conscientes de este problema, los SGBD proporcionan un mecanismo de vistas que permite que cada usuario tenga su propia vista o visión de la base de datos.

Los SGBD están en continua evolución, tratando de satisfacer los requisitos de todo tipo de usuarios. Por ejemplo, muchas aplicaciones de hoy en día necesitan almacenar imágenes, vídeo, sonido, etc.

12.2.2 Tipos de base de datos

Existen muchos tipos diferentes de bases de datos. Algunas de las principales son:

- **Bases de datos relacionales**. La más utilizada actualmente. Las bases de datos relacionales se hicieron predominantes en la década de 1980. Los elementos de una base de datos relacional se organizan como un conjunto

de tablas con columnas y filas. La tecnología de bases de datos relacionales proporciona la forma más eficiente y flexible de acceder a información estructurada.

- **Bases de datos orientadas a objetos**. Una de las que muestra ciertas ventajas con respecto a las BD relacionales. La información de una base de datos orientada a objetos se representa en forma de objetos, como en la programación orientada a objetos.
- **Bases de datos distribuidas**. Una base de datos distribuida consta de dos o más archivos que se encuentran en sitios diferentes. La base de datos puede almacenarse en varios ordenadores, ubicarse en la misma ubicación física o repartirse en diferentes redes.
- **Almacenes de datos**. Un repositorio central de datos, un data warehouse es un tipo de base de datos diseñado específicamente para consultas y análisis rápidos.
- **Bases de datos NoSQL**. Una base de datos NoSQL, o base de datos no relacional, permite almacenar y manipular datos no estructurados y semiestructurados (a diferencia de una base de datos relacional, que define cómo se deben componer todos los datos insertados en la base de datos). Las bases de datos NoSQL se hicieron populares a medida que las aplicaciones web se volvían más comunes y complejas.
- **Bases de datos orientadas a grafos**. Una base de datos orientada a grafos almacena datos relacionados con entidades y las relaciones entre entidades.
- **Bases de datos OLTP**. Una base de datos OLTP es una base de datos rápida y analítica diseñada para que muchos usuarios realicen un gran número de transacciones.

Estos son solo algunos de las varias docenas de tipos de bases de datos que se utilizan hoy en día. Otras bases de datos menos comunes se adaptan a funciones científicas, financieras o de otro tipo muy específicas. Además de los diferentes

tipos de bases de datos, los cambios en los enfoques de desarrollo tecnológico y los avances considerables, como la nube y la automatización, están impulsando a las bases de datos en direcciones completamente nuevas.

12.2.3 Ventajas y desventajas de los SGBD

Los sistemas de bases de datos presentan numerosas ventajas gracias, fundamentalmente, a la integración de datos y a la interfaz común que proporciona el SGBD. Algunas de estas ventajas se describen a continuación:

- **Control sobre la redundancia de datos**. Los sistemas de ficheros almacenan varias copias de los mismos datos en ficheros distintos. Esto hace que se desperdicie espacio de almacenamiento, además de provocar faltas de consistencia de datos (copias que no coinciden). En los sistemas de bases de datos todos estos ficheros están integrados, por lo que no se almacenan varias copias de los mismos datos. Sin embargo, en una base de datos no se puede eliminar la redundancia completamente, ya que en ocasiones es necesaria para modelar las relaciones entre los datos, o bien es necesaria para mejorar las prestaciones.
- **Control sobre la consistencia de datos**. Eliminando o controlando las redundancias de datos se reduce en gran medida el riesgo de que haya inconsistencias. Si un dato está almacenado una sola vez, cualquier actualización se debe realizar sólo una vez, y está disponible para todos los usuarios inmediatamente. Si un dato está duplicado y el sistema conoce esta redundancia, el propio sistema puede encargarse de garantizar que todas las copias se mantengan consistentes.
- **Compartición de datos.** En los sistemas de ficheros, los ficheros pertenecen a los departamentos que los utilizan, pero en los sistemas de bases de datos, la base de datos puede ser compartida por todos los usuarios que estén autorizados.
- **Mejora en la integridad de datos**. La integridad de la base de datos se refiere a la validez de los datos almacenados. Normalmente, la integridad se expresa mediante restricciones o reglas que no se pueden violar. Estas

restricciones se pueden aplicar tanto a los datos, como a sus relaciones, y es el SGBD quien se encargará de mantenerlas.

- **Mejora en la seguridad**. Los SGBD permiten mantener la seguridad mediante el establecimiento de claves para identificar al personal autorizado a utilizar la base de datos.
- **Mejora en la accesibilidad a los datos**. Muchos SGBD proporcionan lenguajes de consulta o generadores de informes que permiten al usuario hacer cualquier tipo de consulta sobre los datos, sin que sea necesario que un programador escriba una aplicación que realice tal tarea.
- **Mejora en los servicios de copias de seguridad y de recuperación ante fallos**. Muchos sistemas de ficheros dejan que sea el usuario quien proporcione las medidas necesarias para proteger los datos ante fallos en el sistema o en las aplicaciones. Los usuarios tienen que hacer copias de seguridad cada día, y si se produce algún fallo, utilizar estas copias para restaurarlos.

La integración de los datos y la existencia del SGBD también plantean ciertos inconvenientes, o desventajas, tales como los que se citan a continuación.

- Alta complejidad. Los SGBD son conjuntos de programas muy complejos con una gran funcionalidad. Es preciso comprender muy bien esta funcionalidad para poder sacar un buen partido de ellos.
- Gran tamaño. Los SGBD son programas complejos y muy extensos que requieren una gran cantidad de espacio en disco y de memoria para trabajar de forma eficiente.
- Costo económico del SGBD. Si bien hay SGBD libres (open source) que ofrecen una gran funcionalidad y muy buenas prestaciones. Hay SGBD de pago, en los cuales el costo varía dependiendo del entorno y de la funcionalidad que ofrece. Además, hay que tener en cuenta el costo de mantenimiento y del personal encargado de la base de datos.

- **Costo del equipamiento adicional**. Tanto el SGBD, como la propia base de datos, pueden hacer que sea necesario adquirir más espacio de almacenamiento. Además, para alcanzar las prestaciones deseadas, es posible que sea necesario adquirir una PC más grande o una máquina que se dedique solamente al SGBD. Todo esto hará que la implantación de un sistema de bases de datos sea más cara.
- **Costo de la conversión**. En algunas ocasiones, el costo del SGBD y el costo del equipo informático que sea necesario adquirir para su buen funcionamiento es insignificante comparado al coste de convertir la aplicación actual en un sistema de bases de datos. Este coste incluye el coste de enseñar a la plantilla a utilizar estos sistemas y, probablemente, el coste del personal especializado para ayudar a realizar la conversión y poner en marcha el sistema. Este coste es una de las razones principales por las que algunas empresas y organizaciones se resisten a cambiar su sistema actual de ficheros por un sistema de bases de datos.
- **Prestaciones.** Un sistema de ficheros está escrito para una aplicación específica, por lo que sus prestaciones suelen ser muy buenas. Sin embargo, los SGBD están escritos para ser más generales y ser útiles en muchas aplicaciones, lo que puede hacer que algunas de ellas no sean tan rápidas como antes.
- **Vulnerable a los fallos**. El hecho de que toda la información esté centralizada en el SGBD hace que el sistema sea más vulnerable ante los fallos que puedan producirse.

12.3 Bases de datos relacionales

El modelo relacional es el modelo de datos en el que se basan la mayoría de los SGBD en uso hoy en día.

En el modelo relacional todos los datos están estructurados a nivel lógico como tablas formadas por filas y columnas, aunque a nivel físico pueden tener una

estructura completamente distinta. Un punto fuerte del modelo relacional es la sencillez de su estructura lógica.

12.3.1 Relaciones

El modelo relacional se basa en el concepto matemático de relación, que gráficamente se representa mediante una tabla.

Una relación es una tabla con columnas y filas. Un SGBD sólo necesita que el usuario pueda percibir la base de datos como un conjunto de tablas. Esta percepción sólo se aplica a la estructura lógica de la base de datos, no se aplica a la estructura física de la base de datos, que se puede implementar con distintas estructuras de almacenamiento a nivel de la programación del SGBD.

Cada fila de la tabla se conoce como un registro o una tupla. Por definición, no hay registros repetidos.

Un atributo es el nombre de una columna de una relación. En el modelo relacional, las relaciones se utilizan para almacenar información sobre los objetos que se representan en la base de datos. Una relación se representa gráficamente como una tabla bidimensional en la que:

- Las filas corresponden a registros individuales.
- Las columnas corresponden a los campos o atributos de esos registros. Los atributos pueden aparecer en la relación en cualquier orden.

Ejemplo

La información de los clientes de una empresa determinada se representa mediante la relación CLIENTES, que tiene columnas para los **atributos** codcli (código del cliente), nombre (nombre y apellidos del cliente), dirección (calle y número donde se ubica el cliente), codpostal (código postal correspondiente a la dirección del cliente) y codciu (código de la ciudad del cliente).

CLIENTES

codcli	nombre	dirección	codpostal	codciu
333	Sos Carretero, Jesús	Mosen Compte, 14	12964	53596
336	Miguel Archilés, Ramón	Bernardo Mundina, 132-5	12652	07766
342	Pinel Huerta, Vicente	Francisco Sempere, 37-10	12112	07766
345	López Botella, Mauro	Avenida del Puerto, 20-1	12439	12309
348	Palau Martínez, Jorge	Raval de Sant Josep, 97-2	12401	12309
354	Murría Vinaiza, José	Ciudadela, 90-18	12990	12309
357	Huguet Peris, Juan Ángel	Calle Mestre Rodrigo, 7	12930	12309

Como podemos observar, la relación anterior tiene 7 registros, y cada registro tiene los atributos codcli, nombre, dirección, codpostal y codciu.

La información sobre las ciudades se representa mediante la relación CIUDADES, que tiene columnas para los atributos codciu (código de la ciudad), nombre (nombre de la ciudad) y codpro (código de la provincia en que se encuentra la ciudad).

CIUDADES

codciu	nombre	codpro
07766	Burriana	12
12309	Castellón	12
17859	Enramona	12
46332	Soneja	12
53596	Vila-real	12

Las anteriores son relaciones que almacenan los datos (y los relacionan) de los clientes y sus ciudades.

12.3.2 Dominio

Un dominio es el conjunto de valores posibles de uno o varios atributos. Los dominios constituyen una poderosa característica del modelo relacional. Cada atributo de una base de datos relacional se define sobre un dominio, pudiendo haber varios atributos definidos sobre el mismo dominio. La siguiente imagen muestra los dominios de los atributos de la relación CLIENTES del ejemplo anterior.

Atributo	*Dominio*	*Descripción*	*Definición*
codcli	codli_dom	Posibles códigos de cliente.	Número hasta 5 dígitos.
nombre	nombre_dom	Nombres de personas: apellido1 apellido2, nombre.	50 caracteres.
dirección	dirección_dom	Domicilios de España: calle, número.	50 caracteres.
codpostal	codpostal_dom	Códigos postales de España.	5 caracteres.
codpue	codpue_dom	Códigos de las poblaciones de España.	5 caracteres.

El concepto de dominio es importante porque permite que el usuario defina, en un lugar común, el significado y la fuente de los valores que los atributos pueden tomar. Esto hace que haya más información disponible para el sistema cuando éste va a ejecutar una operación relacional, de modo que las operaciones que son semánticamente incorrectas, se pueden evitar.

Por ejemplo, no tiene sentido comparar el nombre de una calle con un número de teléfono, aunque los dos atributos sean cadenas de caracteres.

Sin embargo, el importe mensual del alquiler de un inmueble no estará definido sobre el mismo dominio que el número de meses que dura el alquiler, pero sí tiene sentido multiplicar los valores de ambos dominios para averiguar el importe total al que asciende el alquiler.

12.3.3 Claves, clave primaria y clave ajena

Ya que en una relación no hay registros repetidos, estos se pueden distinguir unos de otros, es decir, se pueden identificar de modo único. La forma de identificarlos es mediante los valores de sus atributos. Se denomina superclave a un atributo o conjunto de atributos que identifican de modo único los registros de una relación. Se denomina clave candidata a una superclave en la que ninguno de sus subconjuntos es una superclave de la relación. El atributo o conjunto de atributos K de la relación R es una clave candidata para R si, y sólo si, satisface las siguientes propiedades:

- Unicidad: nunca hay dos tuplas en la relación R con el mismo valor de K.

- Irreducibilidad (minimalidad): ningún subconjunto de K tiene la propiedad de unicidad, es decir, no se pueden eliminar componentes de K sin destruir la unicidad.

El único modo de identificar las claves candidatas es conociendo el significado real de los atributos, ya que esto permite saber si es posible que aparezcan duplicados.

Clave primaria

Se denomina **clave primaria** de una relación a aquella clave candidata que se escoge para identificar sus registros de **modo único**.

Ejemplo

Podemos elegir como la clave primaria de la relación CIUDADES al atributo codciu. En la relación CLIENTES sólo hay una clave candidata que es el atributo codcli, por lo que esta clave candidata es la clave primaria. Están resaltadas en amarillo en las siguientes tablas.

CLIENTES

codcli	nombre	dirección	codpostal	codciu
333	Sos Carretero, Jesús	Mosen Compte, 14	12964	53596
336	Miguel Archilés, Ramón	Bernardo Mundina, 132-5	12652	07766
342	Pinel Huerta, Vicente	Francisco Sempere, 37-10	12112	07766
345	López Botella, Mauro	Avenida del Puerto, 20-1	12439	12309
348	Palau Martínez, Jorge	Raval de Sant Josep, 97-2	12401	12309
354	Murría Vinaiza, José	Ciudadela, 90-18	12990	12309
357	Huguet Peris, Juan Ángel	Calle Mestre Rodrigo, 7	12930	12309

CIUDADES

codciu	nombre	codpro
07766	Burriana	12
12309	Castellón	12
17859	Enramona	12
46332	Soneja	12
53596	Vila-real	12

Clave ajena

Una clave ajena es un atributo o un conjunto de atributos de una relación cuyos valores coinciden con los valores de la clave primaria de alguna otra relación (puede ser la misma). **Las claves ajenas representan relaciones entre**

datos. Por ejemplo, el atributo codciu de CLIENTES relaciona a cada cliente con su ciudad. Este atributo en CLIENTES es una clave ajena cuyos valores hacen referencia al atributo codciu de CIUDADES (su clave primaria). Se dice que un valor de clave ajena representa una referencia al registro que contiene el mismo valor en su clave primaria (registro referenciado). Está indicado en las tablas anteriores con resaltador naranja.

En resumen

Una base de datos relacional está formada por un conjunto de relaciones. A las relaciones se las representa mediante tablas. Cada tabla tiene una serie de columnas (son los atributos). Cada columna tiene un nombre distinto y es de un tipo de datos (entero, real, carácter, fecha, etc.). En las tablas se insertan filas (son los registros), que después se pueden consultar, modificar o borrar.

Cada tabla tiene una clave primaria, que identifica a cada registro de la tabla, y que estará formada por una o varias columnas de esa misma tabla. Sobre las claves primarias se debe hacer respetar una regla de integridad fundamental:

Una clave primaria es una clave irreducible que se utiliza para identificar de modo único las tuplas.

La mayoría de los SGBD relacionales se encargan de hacer respetar esta regla automáticamente.

Ejemplo

En la tabla CLIENTES del ejemplo anterior la clave primaria es “codcli”. Mientras que en la tabla CIUDADES la clave primaria es “codciu”.

Por otra parte, las relaciones entre los datos de distintas tablas se establecen mediante las claves ajenas. Una clave ajena es una columna o un conjunto de columnas de una tabla que hace referencia a la clave primaria de otra tabla (o de ella misma). Para las claves ajenas también se debe cumplir una regla de integridad fundamental:

Si en una relación hay alguna clave ajena, sus valores deben coincidir con valores de la clave primaria a la que hace referencia, o bien, deben ser completamente nulos.

Muchos SGBD relacionales permiten que el usuario establezca las reglas de comportamiento de las claves ajenas que permiten hacer respetar esta regla.

Ejemplo

En la tabla CLIENTES la clave ajena "codciu" hace referencia a la clave primaria de la tabla CIUDADES, y relaciona cada registro de CLIENTES con un registro de CIUDADES.

12.4 Diseño conceptual: El modelo entidad-relación

El primer paso en el diseño de una base de datos es la producción del esquema conceptual. Una metodología para producir estos esquemas es la denominada entidad-relación.

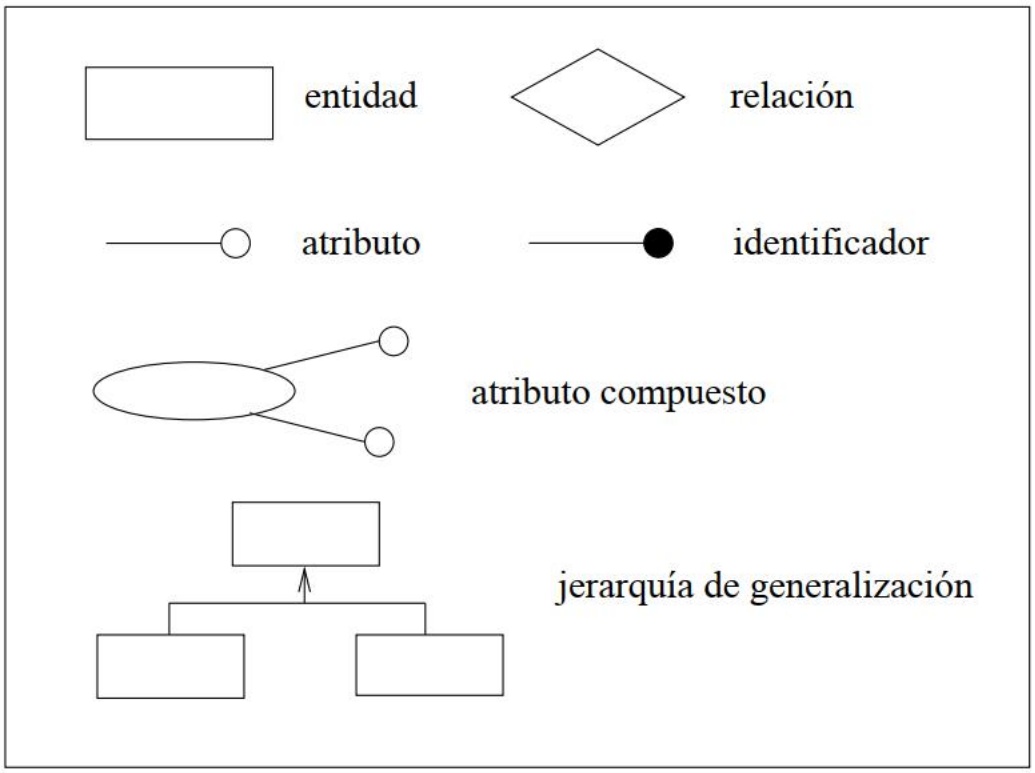
El modelo de datos entidad-relación (E-R) está basado en objetos básicos llamados entidades y en relaciones entre esos objetos.

El modelo E-R se basa hacer corresponder los significados e interacciones del mundo real con un esquema conceptual. Debido a esta utilidad, muchas herramientas de diseño de bases de datos se basan en los conceptos del modelo E-R.

Una entidad es una «cosa» u «objeto» en el mundo real que es distinguible de todos los demás objetos. En el ejemplo que venimos trabajando en esta clase, un cliente y una ciudad son entidades.

El modelo entidad-relación es el modelo conceptual más utilizado para el diseño conceptual de bases de datos. Fue introducido por Peter Chen en 1976. Está formado por un conjunto de conceptos que permiten describir la realidad mediante representaciones gráficas y lingüísticas.

Estos conceptos se muestran en la siguiente figura.



Originalmente, el modelo entidad-relación sólo incluía los conceptos de entidad, relación y atributo. Más tarde, se añadieron otros conceptos, como los atributos compuestos y las jerarquías de generalización, en lo que se ha denominado modelo entidad-relación extendido.

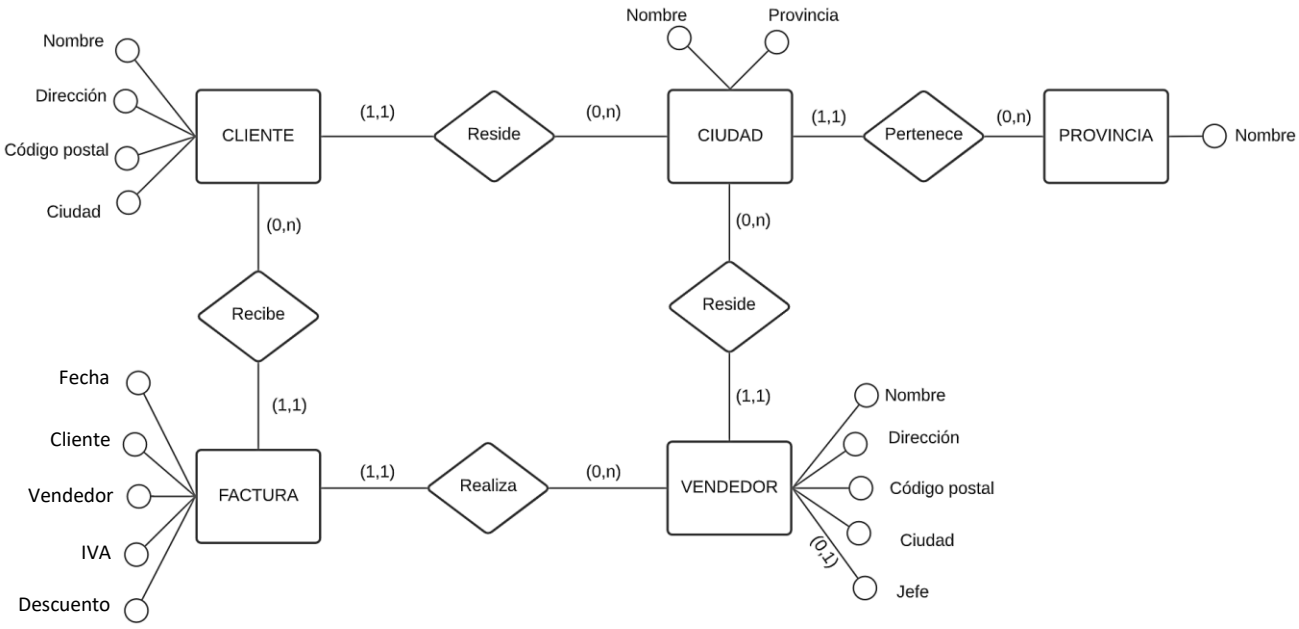
Nosotros vamos a utilizar el modelo que incluye solamente los conceptos de entidad, relación y atributo.

Las tareas a realizar en el diseño conceptual son las siguientes:

1. Identificar las entidades.
2. Identificar las relaciones.
3. Identificar los atributos y asociarlos a entidades y relaciones

Ejemplo

Vamos a ampliar el ejemplo que veníamos trabajando con el siguiente esquema conceptual



Analicemos el esquema:

Entidades: En este caso CLIENTE, VENDEDOR, CIUDAD, PROVINCIA y FACTURA se han representado como entidades porque de ellas se requiere almacenar información: nombre de la ciudad, provincia en la que se encuentra, etc.

Las entidades son las tablas de relación en la base de datos.

Relaciones: Una vez definidas las entidades, se debe definir las relaciones existentes entre ellas. Para identificar las relaciones se suelen buscar las expresiones verbales (reside, realiza, pertenece, etc.). La mayoría de las relaciones son binarias (entre dos entidades), pero también puede haber relaciones en las que participen más de dos entidades.

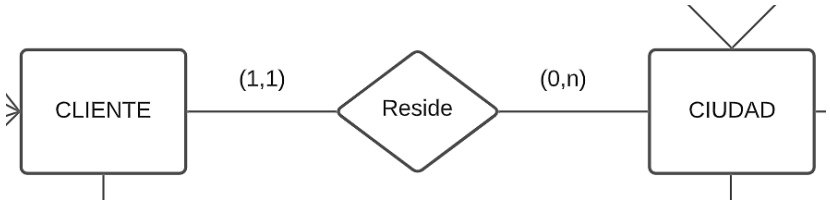
Las relaciones se modelan en la base de datos mediante las **claves ajenas,** que son columnas de las tablas de relaciones.

Una vez identificadas todas las relaciones, hay que determinar la cardinalidad mínima y máxima con la que participa cada entidad en cada una de ellas.

De este modo, el esquema representa de una manera más explícita la semántica de las relaciones. La cardinalidad es un tipo de restricción que se utiliza para comprobar y mantener la calidad de los datos.

La cardinalidad mínima indica si la participación de la entidad en la relación es opcional (se indica con 0) o si es obligatoria (se indica con 1). Que sea obligatoria implica que todas las ocurrencias de la entidad deberán relacionarse con, al menos, una ocurrencia de la entidad que se encuentra al otro lado de la relación. La cardinalidad máxima indica si cada ocurrencia de la entidad sólo puede relacionarse con una ocurrencia de la entidad del otro lado de la relación (se indica con 1), o si puede relacionarse con varias a la vez (se indica con n).

Por ejemplo



La cardinalidad (1,1) de cliente a ciudad indica que cada cliente se relaciona como mínimo con una ciudad y a lo sumo con una ciudad, es decir, con una única ciudad. De cada cliente se sabe que reside en una (1) y solo una ciudad (1), dando lugar a la cardinalidad (1,1).

La cardinalidad (0,n) de ciudad a cliente indica que cada ciudad se relaciona con, como mínimo 0 clientes y como máximo n clientes, es decir, puede haber ciudades que no se relacionen con ningún cliente y ciudades que se relacionen con varios clientes. Una ciudad puede ser lugar de residencia de ningún cliente (0), de un cliente (1) o de varios clientes (n), dando lugar a la cardinalidad (0,n).

Atributos. El siguiente paso consiste en identificar los atributos y asociarlos con las entidades y las relaciones en función de su significado.

Los atributos se modelan mediante las columnas de las tablas de relación de las bases de datos.

Una vez identificados todos los conceptos (entidades, atributos, relaciones, etc.), y dibujado el diagrama entidad-relación, se debe revisar cada esquema conceptual local con los usuarios de las bases de datos, para luego pasar a su implementación.

Una posible implementación del esquema conceptual anterior podría ser:

CLIENTES

codcli	nombre	dirección	codpostal	codciu
333	Sos Carretero, Jesús	Mosen Compte, 14	12964	53596
336	Miguel Archilés, Ramón	Bernardo Mundina, 132-5	12652	07766
342	Pinel Huerta, Vicente	Francisco Sempere, 37-10	12112	07766
345	López Botella, Mauro	Avenida del Puerto, 20-1	12439	12309
348	Palau Martínez, Jorge	Raval de Sant Josep, 97-2	12401	12309
354	Murría Vinaiza, José	Ciudadela, 90-18	12990	12309
357	Huguet Peris, Juan Ángel	Calle Mestre Rodrigo, 7	12930	12309

VENDEDORES

codven	nombre	dirección	codpostal	codciu	codjefe
5	Guillén Vilar, Natalia	Sant Josep, 110	12597	53596	105
105	Poy Omella, Paloma	Sanchis Tarazona, 103-1	12257	46332	
155	Rubert Cano, Diego	Benicarló Residencial, 154	12425	17859	5
455	Agost Tirado, Jorge	Pasaje Peñagolosa, 21-19	12914	53596	5

CIUDADES

codciu	nombre	codpro
07766	Burriana	12
12309	Castellón	12
17859	Enramona	12
46332	Soneja	12
53596	Vila-real	12

PROVINCIAS

codpro	nombre
03	Alicante
12	Castellón
46	Valencia

FACTURAS

codfac	fecha	codcli	codven	iva	dto
6643	16/07/2010	333	105	18	10
6645	16/07/2010	336	105	0	20
6654	31/07/2010	357	155	8	0
6659	08/08/2010	342	5	0	0
6680	10/09/2010	348	455	8	0
6723	06/11/2010	342	5	18	0
6742	17/12/2010	333	105	8	20