

INTRODUCCIÓN A LAS BASES DE DATOS

Un poco de teoría

"Es un error capital teorizar antes de tener información",
Arthur Conan Doyle, Sherlock Holmes, Escándalo en Bohemia.

Se aclara que se ha realizado un intento para no incluir nociones ni terminología que dificulten la lectura de esta primera clase, ya que el único objetivo pedagógico buscado es la introducción al tema.

Se considera que esta clase es fundamental para comprender el modelo que se utiliza en la actualidad para guardar información.

LA INFORMACIÓN COMO RECURSO

Los [datos](#) son observaciones que realizamos en el mundo real.

- Son recopilados como **hechos ó evidencias**
- Adquieren significado a **partir de su procesamiento**

DIFERENCIAS ENTRE DATOS E INFORMACIÓN	
<ul style="list-style-type: none"> • Hay claras diferencias entre datos e información, aunque generalmente se los confunde. Para tratar de entender la diferencia utilizaremos un ejemplo: 	
Dato: 36	Como es simplemente un número, no tiene en sí mismo ningún significado especial. Para que este valor sea de utilidad, es necesario interpretarlo en referencia a algún contexto particular. Como por ejemplo, especificar que ésta es la temperatura máxima registrada en la ciudad de Buenos Aires, en determinado año.
Información: 36°	Temperaturas máxima registradas en la ciudad de Buenos Aires, en el año 1994.
<p>Datos: Los datos son hechos o cosas del mundo, lo suficientemente importantes como para ser registrados. Son recopilados como situaciones o evidencias. Adquieren significado a partir de su procesamiento para convertirse en información. Al hablar de datos, hacemos referencia a un concepto amplio que puede incluir texto, imágenes, sonido, reglas de decisión, etc.</p> <p>Información: Es el resultado del análisis de los datos. Por ejemplo, un satélite colecciona datos (valores numéricos), que una vez volcados en un mapa, son interpretados por un experto que identifica los diferentes usos del suelo en determinada región.</p>	
<ul style="list-style-type: none"> • De este ejemplo podemos inferir que la información es cualitativamente más que los datos y esto es necesariamente así ya que la información debe servirle al usuario para tomar decisiones. Debe tener utilidad, condición que no siempre se cumple con los datos. • Para disponer de información, se requiere de un proceso analítico sobre el dato. • Al contar con Datos confiables y consistentes podemos generar Información que nos permita tomar las decisiones más convenientes. 	
<ul style="list-style-type: none"> • A partir de la información podemos generar Conocimiento, que no es más que: Información + Valor • Hasta llegar al tope de la pirámide de valor, la Sabiduría, estatus en el cual contamos con la información valiosa en el momento, lugar y tiempo adecuados para la generación de ventajas competitivas. 	<p style="text-align: center;">PIRAMIDE DE VALOR</p>

- En las últimas décadas se han realizado importantes avances en la formalización de los modelos que definen la estructura organizativa de los datos, permitiendo su **almacenamiento manipulación y consulta**.
- Estos avances han permitido llegar al concepto de **base de datos relacional**
- Una definición para base de datos...

Una **base de datos** es un conjunto de **información relacionada** que pertenece a una organización y que está **agrupada como un todo**. En la base de datos de una juguetería, por ejemplo, estará reunida la información de los juguetes (precio, cantidad en stock), así como los datos de los proveedores (dirección, teléfono, saldo deudor), clientes (si se desea llevar información individualizada de cada uno de ellos), empleados (salarios, presentismo, comisiones de los vendedores), contabilidad (cobranzas, pagos, liquidaciones), etc.

Tener “agrupada como un todo” esta información trae muchos beneficios, como reducir la **redundancia y la inconsistencia de los datos guardados**.

UN POCO DE TEORÍA

1. HISTORIA

- En los 60, cuando las computadoras empezaron a desarrollarse, la atención estaba centrada en la resolución de problemas particulares: si era necesario procesar información, se programaba especialmente una aplicación particular que solucionaba la cuestión. Un ejemplo típico era la liquidación de sueldos de empleados: una tarea manual muy repetitiva que se automatiza para bajar el costo del procesamiento.
- Como el objetivo era la resolución de problemas, **se prestaba poca consideración al almacenamiento de datos**. Normalmente se archivaban en un formato específico (no-estándar) para cada aplicación. En otras palabras, cada programador trabajaba con un formato de datos propio.
- Este sistema fue adecuado mientras las aplicaciones permanecieron independientes.
- Los **problemas comenzaron cuando fue necesario compartir información** y los formatos que usaban las aplicaciones, no eran compatibles. Le doy un ejemplo: en una empresa había varias aplicaciones que guardaban datos en formatos diferentes. “Compras” mantenía la información sobre los artículos comprados; “Almacén”, sobre los que estaban en stock y “Ventas” sobre los artículos vendidos. Pero, por más que en los tres casos se hablara de los mismos datos (los artículos que la empresa comercializaba), no se podía compartir esa información, ya que las aplicaciones que había desarrollado cada sector, guardaba la información en un formato distinto que servía en forma específica para cada caso.
- Para solucionar este inconveniente, las organizaciones se veían obligadas a **exportar los datos** a los diferentes formatos que requería cada aplicación, tarea que era de por sí bastante engorrosa y no siempre posible.
- Pero el principal problema era que se debía **mantener varias copias** de la misma información en los diferentes formatos que requerían las aplicaciones existentes. Este procedimiento, inevitablemente creaba **redundancia de información**: imaginen en una empresa, la cantidad de veces que era necesario guardar el nombre de cada uno de los productos. A su vez, **traía aparejado inconsistencia**, ya que múltiples copias conllevan inevitablemente a errores. Por supuesto, el costo de mantenimiento era altísimo: si un cliente cambiaba de dirección había que cambiar el dato en tantos archivos como aplicaciones hubiera.
- Cuando la ciencia de la computación y la tecnología avanzaron, se extendió la necesidad de crear un sistema al cual accedieran todas las aplicaciones de la organización. Se pretendía, por ejemplo, que al ingresar los datos de un cliente, este registro sirviera a todos los sectores y que si este cliente cambiaba de domicilio, todas las aplicaciones lo registraran. Por supuesto, esta aspiración era absolutamente legítima e imperiosa.
- En definitiva, se buscaba que todas las aplicaciones en una organización (ventas, contabilidad, compras, etc.) **compartieran un sólo almacén de datos**.
- Es así como nace el **Database Management System (DBMS) o Sistema de Gestión de Base de Datos**, un sistema que servía de proveedor de datos a diversas aplicaciones.
- Este modelo fue propuesto originariamente en 1970 por un matemático de IBM, **Edward Codd**. Debe saber que aunque previamente habían surgido otros modelos (como el modelo jerárquico de datos, o el de red), fueron las “**Bases de Datos Relacionales**” las adoptadas expresamente por casi todos los sistemas comerciales conocidos.
- Gracias a su coherencia y facilidad de uso, el modelo se ha convertido a partir de los años 80 en el más empleado.

El modelo relacional aporta las **siguientes ventajas**:

- **Independencia lógica y física de los datos**, de las aplicaciones.
- **Redundancia mínima**, ya que la base funciona como repositorio común de datos para distintas aplicaciones.
- **Acceso concurrente por parte de múltiples usuarios**

- **Distribución espacial de los datos**, ya que la independencia lógica y física facilita las bases de datos distribuidas. Los datos pueden encontrarse en otra habitación, otro edificio e incluso otro país. El usuario no tiene por qué preocuparse de la localización espacial de los datos a los que accede.
- **Integridad de los datos**, ya que se puede adoptar medidas de seguridad que impidan que se introduzcan datos erróneos. Esto puede suceder tanto por motivos físicos (defectos de hardware, actualización incompleta debido a causas externas), como de operación (introducción de datos incoherentes).
- **Consultas complejas por más de una condición.**
- **Seguridad de acceso y auditoria** Posibilita otorgar derecho de acceso diferenciado a personas y organismos. El sistema de auditoria mantiene el control de acceso a la base de datos, con el objeto de saber qué o quién realizó una determinada modificación y en qué momento.
- **Respaldo y recuperación**, que posibilita al sistema recuperar su estado en un momento previo a la pérdida de datos.
- **Acceso a través de lenguajes de programación estándar** que brinda la posibilidad ya mencionada de acceder a los datos mediante lenguajes de programación ajenos al sistema.

2. TÉRMINOS TÉCNICOS

Es importante que preste especial atención a esta sección, ya que introduce los términos técnicos a los cuales haremos referencia en clase.

- **Database Management System (DBMS) o Sistema de Gestión de Base de Datos** un sistema que sirve de proveedor de datos a diversas aplicaciones.
- **Base de datos:** Una base de datos es un conjunto de información relacionada con un asunto o con una finalidad.
Se compone de **entidades** (cosas u objetos del mundo real distinguibles de todos los demás objetos) que poseen **atributos** (propiedades o características de las que se quiere llevar registro). Las entidades pueden ser cosas concretas, como personas o libros, o abstractas, como un préstamo o una venta.

	<p>Los atributos de la entidad "ALUMNO" podrían ser Documento de Identidad, nombre, apellido, edad, materias cursadas, notas obtenidas, etc.</p>		<p>Los atributos de la entidad "VENTA" podrían ser Fecha, Tipo de Factura, Nº de Factura, Nombre del Vendedor, Nombre del Comprador, Artículos Comprados, etc.</p>
---	--	--	--

- **Identificador o Llave:** Se llama identificador a todo atributo cuyo valor identifica la entidad bajo análisis en forma inequívoca. Si nos encontramos con una entidad que no tiene un atributo que lo identifique, podemos "inventar" un atributo que sirva como identificador.
¿Recuerdan haber realizado alguna vez una consulta en una biblioteca, donde se usaban fichas de cartón? Tiene presente que tenían un número con el que solicitaba el libro al bibliotecario: este número era el Identificador.

Query o consulta: es una **declaración** o grupo de declaraciones en idioma formal que permite comunicarse con la base de datos **para recuperar la información almacenada**, especificando determinadas condiciones.

SQL (Structured Query Language): SQL es un lenguaje para comunicarse con las bases de datos. Creado por IBM a mediados de los setenta. Oracle fue la primera empresa en utilizarlo en una base de datos. Actualmente existe un estándar SQL gestionado por un comité ANSI.

Inconsistencia de datos: utilizaremos un ejemplo en lugar de una definición. Supongamos que una empresa guarda información de clientes. Si existen varios ingresos para cada cliente, se puede generar inconsistencia de información, ya que si este cliente cambia la dirección y nos olvidamos de actualizar la misma en uno de los registros, al realizar una consulta, no se sabría cuál de las direcciones es la correcta.

CASOS

Antes de comenzar, vamos a plantearle 2 casos:

- El caso de una empresa. Utilizaremos los requerimientos de la empresa para explicarle el modelo relacional.
- Por otra parte le planteamos los requerimientos de una escuela primaria que requiere una base de datos para administrar sus operaciones. La resolución de este problema lo ayudará a adquirir el método necesario para crear una base de datos relacional.
-

EMPRESA

Las “reglas de la organización” de la empresa indican que:

- En la empresa hay varios **vendedores**.
- Que cada uno de los vendedores tiene un número de teléfono interno único y exclusivo.
- Que cada vendedor atiende varios **clientes**.
- Que cada cliente puede ser atendido sólo por el vendedor que le ha asignado la empresa.
- Que la empresa comercializa varios **productos**.
- Que cada cliente compra uno ó más productos que la empresa distribuye.

ESCUELA

Las “reglas de la organización” de la escuela indican que:

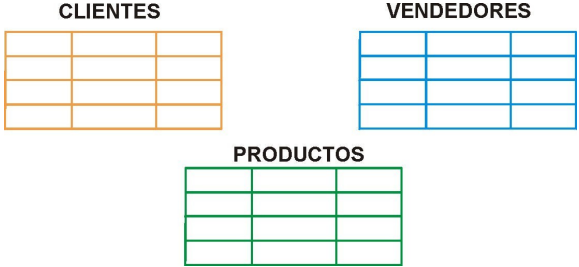
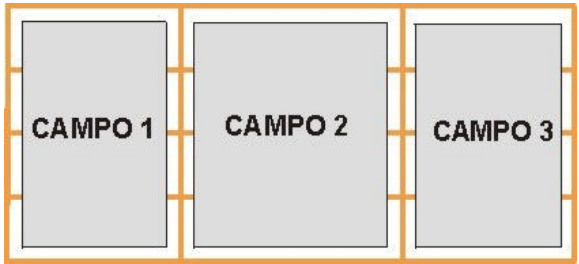

- En la escuela hay muchos **alumnos**
- Cada alumno tiene solo un **maestro**, pero un maestro tiene varios alumnos a cargo
- Cada alumno tiene un solo **responsable**, pero una persona puede ser responsable de más de un alumno a la vez. Un ejemplo típico, un padre con más de un hijo en la escuela.
- Cada responsable a cargo puede tener varios teléfonos. Un teléfono personal, varios laborales, un móvil.
- La escuela ofrece varias **actividades complementarias** optativas, como ser cerámica, francés e inglés. Un alumno puede hacer varias **actividades complementarias**. Obviamente, en cada actividad complementaria estarán inscriptos muchos alumnos.
- Por razones de privacidad, en una tabla aparte se desea tener la información acerca de la **salud del alumno**, como ser su grupo y factor sanguíneo, si tuvo apendicitis, si es alérgico a la penicilina, etc.

3. EL MODELO RELACIONAL DE INFORMACIÓN

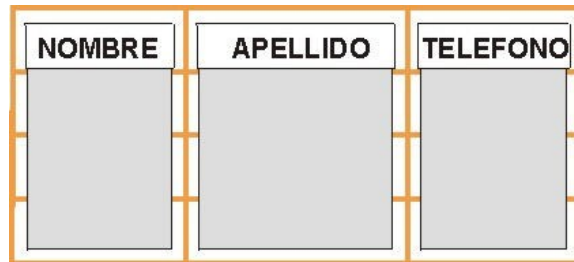
-

3.1. Postulados

El modelo relacional establece los siguientes postulados:

<ul style="list-style-type: none"> • Una base de datos relacional siempre está compuesta por tablas que se conectan (relacionan) entre sí para compartir información. 	
<ul style="list-style-type: none"> • Una tabla es un conjunto de datos sobre una entidad específica: por ejemplo, en una base de datos de una empresa, se guarda en tablas diferentes información sobre los clientes, los vendedores y los productos. 	
<ul style="list-style-type: none"> • Las tablas organizan la información en columnas llamadas campos y filas llamadas registros. 	
<ul style="list-style-type: none"> • Cada registro debe contener información de una solá unidad de análisis: por ejemplo, en la tabla "CLIENTE", una fila por cada cliente. • Hay tantas filas como cliente tenga una empresa. 	
<ul style="list-style-type: none"> • Los campos (columnas) guardan los atributos (propiedades o características) del registro. 	
<ul style="list-style-type: none"> • Los campos pueden ser de distintos tipos: numéricos (edad, cantidad de hijos), alfanuméricos (Nombre, Dirección), fechas (fecha de nacimiento, fecha de ingreso a la compañía), verdadero / falso (Posee auto propio, cumplió el servicio militar), etc. • Todos los datos registrados en una columna deben ser del mismo tipo 	

- El **nombre del campo debe ser único e inequívoco**: no se puede llamar a dos campos diferentes con el mismo nombre.
- Por ejemplo, si se quiere guardar 3 teléfonos por cada cliente, la primera columna se podría llamar Teléfonos_1, la segunda Teléfonos_2 y así sucesivamente.



- ¿Por qué no podemos guardar los 3 teléfonos en una sola columna, como haríamos en una ficha papel?
- Continué leyendo y descubrirá que, en el modelo relacional, las entradas deben ser indivisibles.

- Desarrollemos el ejemplo de una empresa. Analizaremos la información de los clientes.
- La empresa completa una ficha para cada cliente, especificando los datos personales del mismo.
- Esta información se podría guardar en la tabla "CLIENTE", que tendrá tantos registros (filas) como clientes tengan la organización.
- Cada registro se compondrá de un cierto número de campos (columnas), como ser N° de Documento, Nombre, Apellido, Dirección, Teléfono.

CLIENTE					
N° de Cliente	N° de Documento	Nombre	Apellido	Dirección	Teléfono
0023	10.551.985	Juana	Molina	Cabildo 110	4-784-6789
0056	11.346.897	Alberto	Alaluf	Lima 345	4-678-6782
0067	12.234.674	Maria	Jurez	Paz 897	3-457-8764
0098	12.456.432	Pedro	Colombo	Bolivia 877	6-567-8908

- Así como cuando se organiza una biblioteca, se crea un índice que permite hacer una referencia a un libro fichado, en las bases de datos es necesario la creación de un campo denominado **identificador** que permita localizar la unidad de análisis. En este caso el cliente.
- En ningún momento dos filas de la misma tabla pueden tener el mismo identificador.
- En este caso, N° de Cliente es el atributo identificador que permite reconocer al comerciante en forma inequívoca.

- El modelo relacional introduce el concepto de **valor nulo**, para hacer alusión a valores que **no son todavía conocidos**. Por ejemplo, en la tabla clientes, la columna estatura puede estar vacía para determinado registro, no indicando esto que no tenga altura, sino que no se conoce la misma.

- En este ejemplo se ve claramente la diferencia entre “cero” y “nulo”.
- ¿Por qué el concepto de valor nulo es tan importante? Porque los registros con valores nulos **se excluyen de las estadísticas**.

Promedio estatura de clientes: 1.16 (un disparate)
Promedio considerando valores nulos: 1.74

NOMBRE	ESTATURA
CLIENTE 1	1.73
CLIENTE 2	
CLIENTE 3	1.84
CLIENTE 4	1.75
CLIENTE 5	
CLIENTE 6	1.63

- **El orden de las columnas y las filas no tiene importancia**. Se podrá recuperar la información en el orden que se desee, sin importar la disposición en que fue ingresada.
- Las entradas de información **deben ser atómicas**. En otras palabras, debe ser información que no pueda ser divisible. Por ejemplo, en una ficha papel es admisible registrar dos idiomas (inglés, francés) en un mismo casillero, pero en el modelo relacional esto no es admisible. Una posible solución, aunque altamente desaconsejable, sería la creación de diferentes columnas: idioma 1, idioma 2, etc.
- ¿Por qué es desaconsejable? Porque si alguna persona no habla dos idiomas, estaremos desperdiciando espacio (creamos una columna que no tendrá datos). Y si habla más de dos... ¿Dónde lo escribimos? Este problema se denomina **“de tributos repetidos o de grupos repetitivos”**. Volveremos sobre este tema y daremos una solución más adelante en esta misma clase.
- ¿Recuerda el problema que se presentaba para guardar varios teléfonos por cliente? También es un típico ejemplo de atributos repetidos.

3.2. Relaciones entre tablas

- Si ahora agregáramos en la tabla “CLIENTE”, al VENDEDOR que atiende a cada uno de ellos, con todos sus datos personales, surgiría una enorme cantidad de problemas. Además de tener que ingresar todos los datos del vendedor por cada cliente (**redundancia**), si algún empleado cambia de teléfono deberíamos actualizar la información en muchos registros, tantos como clientes tenga a cargo el vendedor, lo que llevaría indefectiblemente a generar **inconsistencias** (que en un registro figure el teléfono viejo y en otro el nuevo).
- También se agrandaría **el tamaño de la base**, lo que haría más costoso el medio de almacenamiento (discos rígidos) y más largas las búsquedas (porque se necesita pasar por registros más largos).
- Todos estos problemas se solucionan si se crea una tabla llamada “**VENDEDORES**”, donde cada registro corresponda a un empleado de la empresa del sector ventas.
- Nuevamente, nos debemos asegurar que cada registro tenga **su identificador único** (podría ser el legajo interno de la empresa).

VENDEDORES					
Nº de Vendedor	Nº de Documento	Nombre	Apellido	Sector	Interno
1	14.985.551	Juan	Aragón	Agroquímicos	23
2	11.897.346	Manuela	Sarquis	Fertilizantes	24

- Por otra parte, se debería **relacionar esta nueva tabla con la tabla CLIENTE**.



- Para relacionar la tabla VENDEDOR con la tabla CLIENTE, en la tabla CLIENTE creamos un campo donde se ubica el identificador del VENDEDOR que nos remitirá a los datos personales del mismo.
- A partir de esta relación, obtenemos para cada cliente, los datos del vendedor que le ha sido asignado.

VENDEDORES					
Nº de Vendedor	Nº de Documento	Nombre	Apellido	Sector	Interno
1	14.985.551	Juan	Aragón	Agroquímicos	23
2	11.346.897	Manuela	Sarquis	Fertilizantes	24

CLIENTES						
Nº de Cliente	Nº de Documento	Nombre	Apellido	Dirección	Teléfono	Nº de Vendedor
0023	10.551.985	Juana	Molina	Calles 110	4-784-8789	1
0056	11.346.897	Alberto	Alaruf	Lima 345	4-678-8782	1
0067	12.234.674	Maria	Jurez	Paz 897	3-457-8764	1
0098	12.456.432	Pedro	Colombo	Bolivia 877	6-567-8908	2

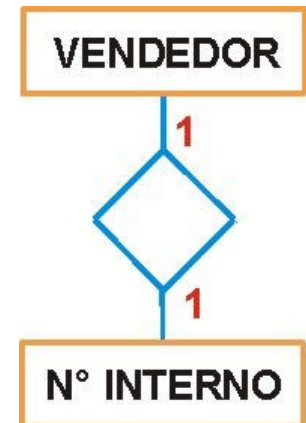
- Vamos a probar si funciona. ¿Sabe cuál es el nombre y apellido del vendedor que atiende al cliente 0023?
- Resumiendo, en general en una base de datos **hay más de una tabla**, que habitualmente están relacionadas entre ellas.
- Las tablas se conectan a través de un campo común que ambas poseen: específicamente el **identificador** de una tabla puede ser **atributo de otra** con la finalidad de **proveer un vínculo entre ambas**.
- Ese fue el caso de CLIENTE y VENDEDORES. El identificador primario de "CLIENTE" era Nº de Cliente, el identificador primario de "VENDEDORES" era el Nº de Vendedor y se relacionaban mediante el campo Nº de Vendedor.
- En la tabla CLIENTE, al campo Nº de Vendedor, no es el identificador primario: **es el identificador secundario**.
- Recapitulando, se llama:
 - **Identificador Primario** (o llave primaria): El campo que identifica inequívocamente a una entidad.
 - **Identificador Secundario** (o llave foránea): El campo de una tabla que está vinculado con el identificador primario de otra tabla, creando así una relación.

3.3. Tipo de relaciones

Son varios los tipos de relaciones que se pueden entablar entre dos tablas. Los analizaremos a continuación.

UNO A UNO:

- Si cada unidad de la tabla A **está relacionada con una sola unidad de la tabla B y viceversa**, entonces la relación es Uno-a-Uno.
- En este ejemplo, cada vendedor tiene un sólo número telefónico interno asignado y cada número de interno es asignado a un sólo vendedor (la relación –rombo- se denominaría “tiene asignado” es del tipo uno a uno).
- Resumiendo: cada vendedor tiene un número de interno exclusivo que no comparte.



- En este caso los **identificadores primarios coincidentes** permiten cruzar información entre tablas.

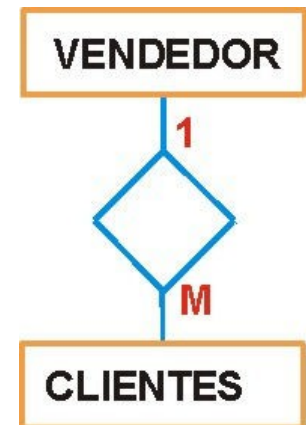
NOMBRE	IDENTIFICADOR	DIRECCION
	VENDEDOR 1	
	VENDEDOR 2	
	VENDEDOR 3	

INTERNO	IDENTIFICADOR
101	VENDEDOR 1
102	VENDEDOR 2
103	VENDEDOR 3

- Por esta no es la única posibilidad de utilizar tablas cuyas relaciones es uno a uno. Este tipo de relación se utiliza frecuentemente para guardar **información diferenciando entre datos frecuentes, de ocasionales**. En el párrafo siguiente desarrollaremos este concepto utilizando un ejemplo.
- En una tabla de VENEDORES, se podrían archivar los datos usados frecuentemente, como: nombre, puesto, salario base, dirección, teléfono, fecha de nacimiento. En otra tabla, se podrían almacenar aquellos datos usados en forma excepcional como gustos musicales, pasatiempos, nombre, fecha de nacimiento y ocupación del cónyuge, nombres y fechas de nacimiento de los hijos, nombres de sus mascotas, auto que posee.

UNO A MUCHOS:

- Si una unidad cualquiera de la tabla **A** **está relacionada con varias de la tabla B**, pero cada unidad de **B** **está en relación con sólo una unidad de A**, se dice que la relación es Uno-a-Muchos.
- En el ejemplo de la empresa, cada vendedor atiende a varios clientes, pero cada cliente sólo puede consultar al vendedor que se le ha asignado (la relación se podría llamar "atiende").



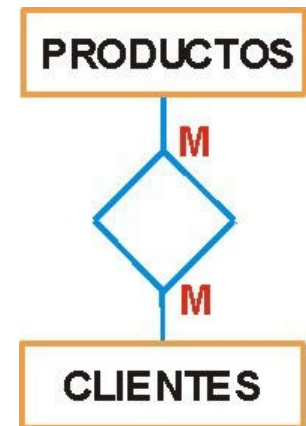
- En este caso, las tablas se conectan a través de un campo común que ambas poseen: específicamente el **identificador** de una tabla puede ser el **atributo de otra**, con la finalidad de proveer un vínculo entre ambas.
- En este caso, el Identificador del vendedor será el Identificador Primario de la tabla VENDEDOR e identificador secundario de la tabla CLIENTE.

NOMBRE	VENDEDORES	DIRECCION
	VENDEDOR 1	
	VENDEDOR 2	
	VENDEDOR 3	

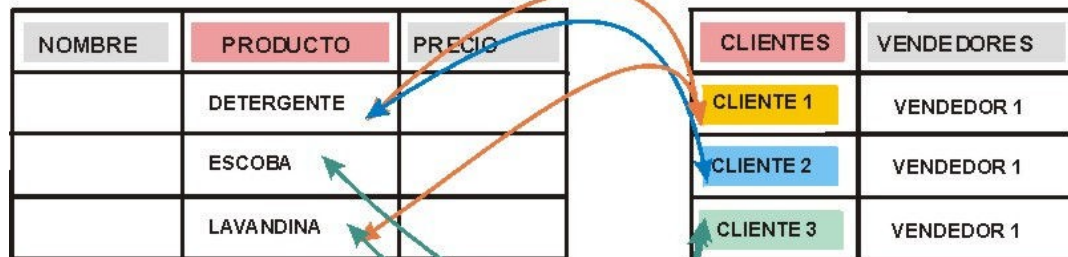
CLIENTES	VENDEDORES
CLIENTE 1	VENDEDOR 1
CLIENTE 2	VENDEDOR 1
CLIENTE 3	VENDEDOR 1

MUCHOS A MUCHOS:

- Si una unidad cualquiera de la tabla **A**, **está relacionada con varias de la tabla B** y una unidad cualquiera de **B**, **está en relación con varias de la tabla A**, se dice que la relación es Muchos-a-Muchos.
- En el ejemplo de la empresa, son varios los clientes y varios los productos. Por este motivo, un cliente puede comprar varios productos, pero a su vez cada producto puede ser adquirido por varios clientes (se podría llamar a la relación "compra").



- Por ejemplo, el cliente 1 compra detergente y lavandina; mientras que el cliente 2, sólo detergente; y el cliente 3, lavandina y escobas.
- En el modelo relacional, este tipo de relación **no está permitida**.
- Sin embargo, el modelo ofrece soluciones para estos casos, que son tan frecuentes en la realidad.



3.4. Integridad Relacional

- Ahora que ya conoce el funcionamiento de las claves primarias y las claves secundarias esta en condiciones de estudiar las **reglas de integridad**.
- Con este nombre se designa aquellas reglas que han de ser aplicadas a una base de datos para asegurar que los datos introducidos **sean consistentes con la realidad que se pretenden modelar**.
- Existen dos reglas generales que aporta el modelo relacional. Estas dos reglas son muy simples:

Primera Regla:

Regla de integridad de las entidades: ningún componente de la clave primaria de una relación base puede aceptar valores nulos.

- Esta regla impide la existencia de un registro sin identificador único.

Segunda Regla:



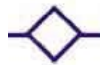



Regla de integridad referencial: la base de datos no debe contener valores de clave ajena o secundaria, sin concordancia con una clave principal.

- Esta regla impide que, por ejemplo, en una base de una escuela, exista un alumno con un responsable inexistente, o un responsable sin alumnos a cargo.
- Además, muchos RDBMSs añaden un buen número de características que ayudan a mantener más fácilmente la integridad de los datos. Mediante estos mecanismos es posible añadir reglas específicas para cada tabla en una base de datos; éstas son las denominadas restricciones de integridad definidas por el usuario. Por ejemplo, podríamos determinar que un alumno no pueda ser mayor de X años o que un arancel de un curso no pueda ser un valor negativo. El resultado sería que al intentar introducir un valor fuera de este rango, el DBMS rechazaría la información introducida mostrando un mensaje de error.

3.5. Vistas y Tablas

<ul style="list-style-type: none"> • En el modelo relacional se pueden distinguir distintos perfiles de usuarios. Por ejemplo: (Operador, Administrador, Gerente). A cada usuario, se le puede asignar un tipo distinto de permiso: sólo lectura, lectura y escritura pero sólo de ciertas tablas, etc... • Con este objetivo, las tablas pueden ser de dos tipos: tablas y vistas.
<ul style="list-style-type: none"> • Las tablas son reales en el sentido que contienen la información.
<ul style="list-style-type: none"> • Las vistas son tablas virtuales. Se construyen a partir de tablas, siendo una combinación de filas y columnas de una o más tablas. El objetivo de las mismas es presentar la información a un particular usuario, en la forma que es útil para su propósito. Una vista puede ocultar los datos que el usuario no necesita o no debe ver.
<ul style="list-style-type: none"> • Supongamos que en un banco, hay una Base de Datos con las siguientes tablas: CLIENTE (<u>Nº de Cliente</u>, Nombre, Apellido, Nº de Préstamo, etc.) PRÉSTAMO (<u>Nº de Préstamo</u>, Nº de Sucursal, Monto, Cantidad Cuotas, etc.) • Un empleado a nivel gerencial necesita conocer toda la información disponible de los préstamos otorgados. • Sin embargo, los empleados no están autorizados a ver la información concerniente a los préstamos concretos que pueda tener cada cliente (monto, cuotas, etc.) • Por lo tanto, se le debe negar el acceso a la información fundamental de la tabla Préstamo. Pero se puede crear una Vista que muestre solamente si el cliente tiene o no tiene préstamos.

4. MODELO EAR (ENTIDAD-ATRIBUTO-RELACIÓN)

<ul style="list-style-type: none"> • Vamos a introducir a continuación un modelo que le facilitará la construcción de la base de datos. • Este modelo fue creado en 1976 por Peter Chen del "Massachusetts Institute of Technology" y es ampliamente usado como un medio de modelado de bases de datos. 	
<ul style="list-style-type: none"> • En el modelo Entidad/Relación, la realidad se representa mediante un número muy reducido de conceptos semánticos básicos, que detallaremos a continuación. 	
<ul style="list-style-type: none"> • El mundo está compuesto de entidades. Una entidad es cualquier objeto distinguible y relevante en el mundo en cuestión (los clientes de nuestro ejemplo anterior). • Las entidades poseen un número indeterminado de propiedades que son "trozos" de información que describen a las entidades de uno u otro modo. • Cada una de las entidades tiene un identificador, en otras palabras son identificables de forma única e inequívoca. • Grupos de entidades conectadas mantienen relaciones con otros grupos de entidades. • Se supone que mediante estos simples componentes se puede modelar cualquier "sección de realidad". 	
<ul style="list-style-type: none"> • El modelo EAR aporta una herramienta de modelado para representar las entidades, propiedades y relaciones: los diagramas Entidad/Relación. • Mediante éstos, el esquema conceptual abstracto puede ser graficado, logrando mantener una independencia conceptual con respecto a la implementación propiamente dicha. • En realidad, podemos hacer que los diagramas sean un fiel reflejo de las relaciones, interrelaciones y atributos del modelo relacional de datos. 	
<ul style="list-style-type: none"> • En estos diagramas, las entidades se dibujan como rectángulos 	
<ul style="list-style-type: none"> • Sus atributos, como elipses atadas a los rectángulos por líneas 	
<ul style="list-style-type: none"> • Las relaciones como rombos 	
<ul style="list-style-type: none"> • El tipo de relación entre dos entidades se representa mediante la convención que se muestran en la imagen: Uno-a-Muchos en este ejemplo. 	
<ul style="list-style-type: none"> • Una propiedad cuyo nombre está subrayado, es la clave primaria ó identificador. 	
<ul style="list-style-type: none"> • Finalmente, un rectángulo doble, significa que esa entidad es dependiente o débil, es decir, su existencia depende de la existencia de otra entidad. Por ejemplo, se podría establecer la restricción de que un teléfono no pudiera existir sin estar adscrito a un cliente. 	
<ul style="list-style-type: none"> • Este modelo incluye 4 pasos. Los ejemplificaremos usando el modelo de datos de la una empresa. 	

1^{er} PASO: IDENTIFICAR LAS ENTIDADES

- Las entidades son objetos o cosas que pueden identificarse y tener una existencia independiente y sobre las cuales la organización necesita información.
- Son en general **sustantivos**. En nuestro ejemplo, luego de un primer tanteo en la empresa, encontramos las siguientes entidades: los CLIENTE, cada uno de los cuales es atendido por un VENDEDOR; cada vendedor tiene un N° de teléfono INTERNO; los clientes compraban diversos PRODUCTOS.

- Cada una de las entidades es representada con **un rectángulo** que contienen el nombre de la entidad.

CLIENTES

- En este ejemplo hay 4 entidades:

CLIENTE, VENDEDOR, INTERNO, PRODUCTO

- Por cada entidad se crea una tabla.

2^{DO} PASO: IDENTIFICAR LAS RELACIONES ENTRE TABLAS ESPECIFICANDO TIPO DE RELACIÓN

- Una relación es **una asociación entre dos objetos**. Las relaciones son representadas por **un rombo** entre los rectángulos. A veces se coloca el nombre de la relación dentro del rombo.
- En este momento es importante **reconocer los tipos de relaciones**. Las posibilidades son:
 Uno-a-Uno
 Uno-a-Muchos
 Muchos-a-Muchos

VENDEDOR**1****CLIENTES**

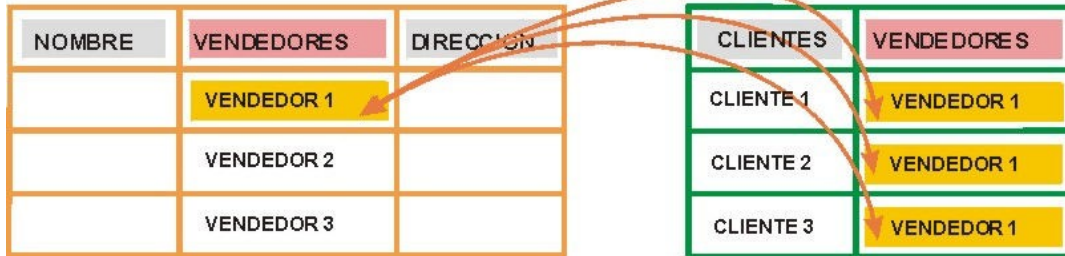
- Afortunadamente, las reglas para transformar las relaciones del diagrama, en relaciones del modelo relacional, son bastante directas y simples.
- Vamos a analizarlas a continuación.

Uno-a-Uno:

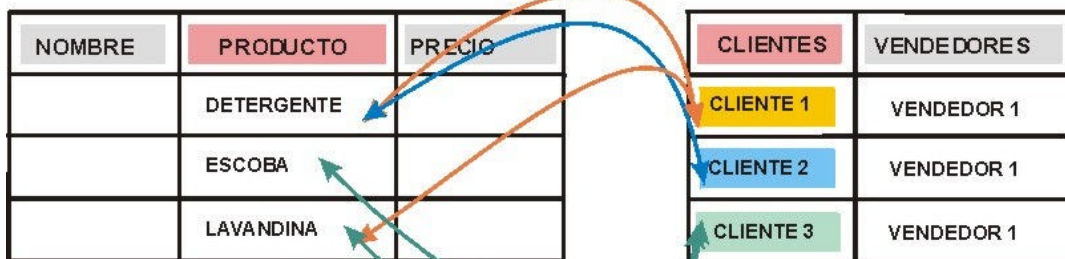
- En el ejemplo de la empresa, VENDEDOR e INTERNO mantienen relaciones Uno-a-Uno.
- Si la relación es Uno-a-Uno, es aceptable **unir estas entidades** en una sola tabla.
- En este caso se podría considerar al N° de Teléfono interno como un atributo del vendedor.
- Alternativamente, **podría ser más conveniente guardar las dos entidades separadas en dos tablas**, usando el identificador común para cruzar información entre tablas. En este ejemplo, ambas tablas podrían estar unidas por el identificador "Vendedor". En la tabla VENDEDOR, este campo será clave primaria y en la tabla INTERNO, clave secundaria.

Uno-a-Muchos:

- En el caso, VENDEDOR y CLIENTE mantienen relaciones Uno-a-Muchos.
- La relación Uno-a-Muchos **se resuelve agregando la llave primaria de la tabla de Uno (VENDEDOR) a la tabla de Muchos (CLIENTE).**
- Este campo, automáticamente se convierte en **llave secundaria** en la tabla CLIENTE.

**Muchos-a-Muchos:**

- En el caso, CLIENTE y PRODUCTO mantienen relaciones Muchos-a-Muchos.
- Por ejemplo, varios clientes pueden comprar un producto y cada producto puede ser adquirido por varios clientes (el cliente 1 compra detergente y lavandina, mientras que el cliente 2, sólo detergente y el cliente 3, lavandina y escobas)
- Las relaciones Muchos-a-Muchos **no son aceptadas por el sistema.**



En este caso se debe pasar al 4^{TO} PASO: se deben derivar nuevas tablas.

3^{ER} PASO: IDENTIFICAR LOS ATRIBUTOS DE LAS ENTIDADES

- En este paso se asignan atributos a las entidades.
- Los atributos son en general **Adjetivos**, características de las entidades que queremos registrar. En el diagrama se muestran como **elipses** atadas a los rectángulos por líneas.
- El sentido común nos dice que los atributos elegidos dependerán del uso que se dará a la base de datos.

CLIENTES

Nombre

Dirección

- De una misma persona, puede interesarnos grupo y factor sanguíneo, si estamos diseñando el modelo de datos de un Hospital, o la carrera y promedio de notas, si el diseño es para una Universidad.
- En este momento, es fundamental prestar especial atención a los **identificadores**. En una tabla donde no existe identificador, se debe crear uno artificial, que en la mayoría de los casos no será nada más que un número único (Número de... o Código de... son los más comunes).

4^{TO} PASO: DERIVAR NUEVAS TABLAS

- El proceso esencial para organizar datos en tablas relacionales, es llamado **Normalización**.
- Consiste en convertir relaciones complejas (Muchos-a-Muchos) en otras más simples que cumplan las condiciones del modelo relacional.

Muchos a Muchos:

- Si la relación es Muchos-a-Muchos, el modelo propone la descomposición de esta relación en dos relaciones Uno-a-Muchos, usando las llaves primarias de las tablas que se quieren relacionar, como atributo en la tabla de conexión.
- Recuerda: ¿CLIENTE y PRODUCTO mantenían relaciones Muchos-a-Muchos?
- En este caso, **se crea una tabla de conexión** que llamaremos COMPRAN. La llave primaria de esta tabla estará compuesta por las **dos llaves primarias de las respectivas tablas**, la de la CLIENTE y la de PRODUCTO.
- La combinación **de ambas llaves constituye la llave principal**.



- Esta tabla de conexión intermedia, posibilita convertir una relación Muchos-a-Muchos en relaciones **Uno-a Muchos y Muchos-a-Uno**.

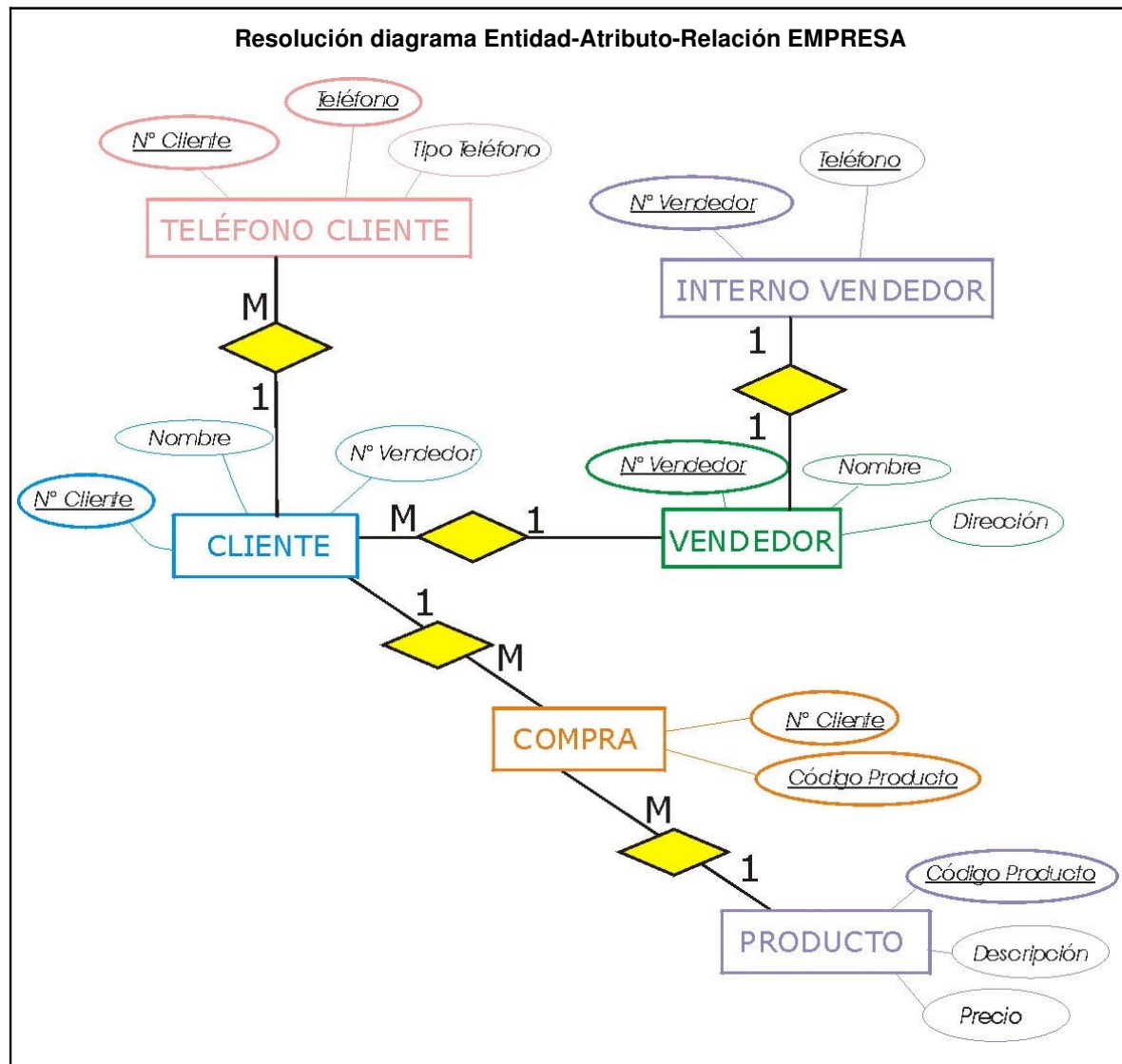
PRODUCTO	PRECIO
DETERGENTE	3.2
ESCOBA	5.9
LAVANDINA	1.6

CANT	PRODUCTO	CLIENTES
6	DETERGENTE	CLIENTE 1
10	LAVANDINA	CLIENTE 1
520	DETERGENTE	CLIENTE 2
112	ESCOBA	CLIENTE 3
11	LAVANDINA	CLIENTE 3

CLIENTES	VENDEDORES
CLIENTE 1	VENDEDOR 1
CLIENTE 2	VENDEDOR 1
CLIENTE 3	VENDEDOR 1

- Pero ese no es el único caso en que se usa la **normalización**. También se la utiliza para ofrecer una solución al problema denominado **de atributos repetidos**. Por ejemplo, el caso de querer almacenar para cada cliente varios números de teléfono. Tal vez se pregunte ¿Por qué no puedo guardar todos los números en una sola columna, como haría en una ficha papel? Porque el modelo relacional estipula que las entradas deben **ser atómicas**.
- Ante esta restricción, la solución más fácil sería crear tantas columnas como teléfonos pudiera tener un cliente. De esta forma, se crearían las columnas **Teléfono_1, Teléfono_2, Teléfono_3, Teléfono_4** y así sucesivamente.
- Esta solución es totalmente aceptable. ¿Pero que pasaría con un cliente que tuviera 50 teléfonos? ¿Crearía 50 columnas?
- Resumiendo: el problema con esta solución es que crecería en forma desproporcionada la base de datos, ya que habría un montón de columnas que casi no se utilizarían.
- Esta es la solución para este problema que propone la normalización:

TELÉFONO	CLIENTES	CLIENTES	VENDEDORES
4-785-6652	CLIENTE 1	CLIENTE 1	VENDEDOR 1
4-785-5636	CLIENTE 1	CLIENTE 2	VENDEDOR 1
4-544-6963	CLIENTE 2	CLIENTE 3	VENDEDOR 1
4-256-8555	CLIENTE 3		
4-256-9632	CLIENTE 3		



5. VENTAJAS Y DESVENTAJAS DEL MODELO RELACIONAL

5.1 Ventajas

- El modelo relacional **es eficiente en la** organización y manejo de grandes colecciones de datos.
- **Permite restricciones de seguridad**, distinguiendo distintos perfiles de usuarios (operador, administrador, Gerente) y asignándole a cada uno de ellos distintos tipos de permisos (sólo lectura, lectura y escritura pero sólo de ciertas tablas, etc.)
- **Posibilitan múltiples vistas de una misma base** según los requerimientos de información del sistema. ¿Recuerda el ejemplo del empleado bancario que no debía acceder a la información de los préstamos?
- **Posibilitan las bases de datos distribuidas**: bases divididas en partes que residen en lugares geográficamente distantes, pero que están fuertemente vinculadas funcionando como una base de datos.
- **Reducen los datos duplicados y redundantes evitando inconsistencias**.
- **Mantienen la integridad y calidad del sistema, ya que los datos están separados de las aplicaciones**. Toda lectura o escritura debe pasar por el DBMS.
- **Permiten el acceso de varios usuarios en el mismo momento**.
- **No requieren conocimientos de programación para acceder a los datos**.

5.2 Desventajas

- **No soportan las relaciones complejas que existen en el mundo real** Por ejemplo, objetos conformados de otros objetos (ejemplo: objeto “propiedad inmueble” compuesto de los objetos “edificio”, “terreno” y “cerca”).
- **No tienen conocimiento semántico**, es decir, no entienden qué significan los datos que guardan. Por ejemplo, las bases de datos con información hidrológica no “saben” que el agua fluye en declive.
- **Los tipos de datos son limitados** (numéricos, alfanuméricos, etc.) No reconocen tipos de datos más complejos. Para dar un ejemplo trivial, sería bueno que fueran capaces de reconocer un dato de tipo “Votante” y que este tipo de dato seleccionará de una base de personas a los individuos mayores de 18 años de edad.
- **Tienen dificultades con el tiempo considerado como una sucesión natural** Una secuencia de hechos, donde los eventos pasan antes o después de otros, es muy difícil de modelar en una base de datos relacional. En general, sólo se obtiene un registro de la información en determinado momento. La consulta del “antes” es siempre difícil.