

Base de Datos I

I. CONCEPTOS SOBRE BASE DE DATOS



3.1. Tipos de datos SQL - Introducción

Se conoce como dato a cualquier elemento que tenga relevancia para un usuario. La informática se ha encargado de proporcionar herramientas que faciliten la manipulación de los datos.

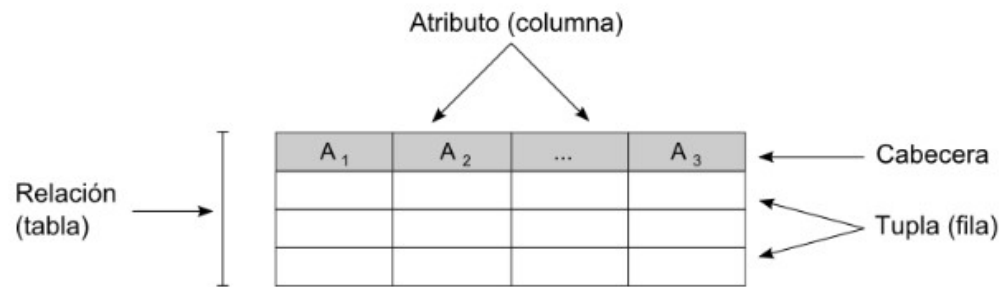
Antes de la aparición de las aplicaciones informáticas, las empresas tenían como únicas herramientas de gestión de datos los ficheros, carpetas y fichas, libros.

En este proceso manual, el tiempo requerido para manipular estos datos era enorme. Pero la propia informática ha evolucionado adaptando sus herramientas para que los elementos que el usuario utiliza en cuanto a manejo de datos se parezcan a los manuales.

Una agenda con los nombres y teléfonos de un conjunto de personas conocidas es una base de datos, puesto que es una colección de datos relacionados con un significado implícito..

Significado implícito de los datos que se atribuye dependiendo del contexto en que se utilizan los mismos. Por ejemplo, el dato fecha en una base de datos de VENTAS puede referirse a la fecha de emisión de las facturas, mientras que si la base de datos es de MÚSICA quizás corresponda a la fecha en que se grabó un tema musical. Es decir, el significado de un dato, depende de la BD que lo contenga

3.1. Tipos de datos SQL - Introducción



Elementos del modelo relacional.

En una relación es importante que cada atributo sea del tipo correspondiente a la columna a la que pertenece. Es decir, que sea coherente con el esquema.

PERSONAS(DNI, Nombre, Altura, Edad, Ciudad)

Una Base de Datos contiene normalmente más de una tabla, ya que suelen ser muchos los tipos de datos a almacenar y resulta conveniente dividirlos en distintas tablas. Además es necesario definir relaciones entre las distintas tablas, y para ello se emplean los denominados atributos *clave*

3.1. Tipos de datos SQL

Cuando se crea una asignación se crea un conjunto de instrucciones para el servicio de integración de datos para que lea los datos de un origen, los transforme y los escriba en un destino.

El servicio de integración de datos transforma los datos según un flujo de datos en la asignación, comenzando por la primera transformación de la asignación, y según el tipo de datos asignado a cada puerto en una asignación.

Developer Tool muestra dos tipos de datos:

Tipos de datos native:

Los tipos de datos nativos son específicos para la tabla relacional o para el archivo sin formato que se utiliza como un objeto de datos físicos. Los tipos de datos nativos aparecen en las propiedades de columna del objeto de datos físicos.

3.1. Tipos de datos SQL

Tipos de datos de transformación:

Son conjuntos de tipos de datos que aparecen en las transformaciones, son tipos de datos internos basados en tipos de datos genéricos ANSI SQL-92, que el servicio de integración de datos utiliza para mover los datos de una plataforma a otra.

Los tipos de datos de transformación aparecen en todas las transformaciones de una asignación, estos incluyen los siguientes tipos de datos:

- Tipo de datos primitivo: Representa un único valor de datos en una sola posición de columna.
- Tipo de datos complejos: Representa varios valores de datos en una sola posición de columna. Utilice tipos de datos complejos en las asignaciones que se ejecutan en el motor de Spark para procesar datos jerárquicos en archivos complejos.

Cuando el servicio de integración de datos lee datos de un origen, convierte los tipos de datos nativos en tipos de datos equivalentes antes de transformar los datos. Cuando el servicio de integración de datos escribe en un destino, convierte los tipos de datos de la transformación en tipos de datos nativos equivalentes.

3.1. Tipos de datos SQL

Los diferentes tipos de datos SQL se utilizan para **definir la estructura de las tablas de la base de datos y para almacenar los datos**. Aunque hay otros tipos de datos, como el de fecha/hora o los datos Booleanos, que generalmente son de un manejo específico o particular

Tipos de datos numéricos

Tipo de datos	Descripción	Almacenamiento
bit	Entero que puede ser 0, 1 o NULL	
tinyint	Permite números enteros de 0 a 255	1 byte
smallint	Permite números enteros entre -32,768 y 32,767	2 bytes
int	Permite números enteros entre -2,147,483,648 y 2,147,483,647	4 bytes
bigint	Permite números enteros entre -9,223,372,036,854,775,808 y 9,223,372,036,854,775,807	8 bytes
decimal (p, s)	Números de escala y precisión fijos. Permite números de $-10^{38} + 1$ a $10^{38} - 1$. El parámetro p indica el número total máximo de dígitos que se pueden almacenar (tanto a la izquierda como a la derecha del punto decimal). p debe ser un valor de 1 a 38. El valor predeterminado es 18. El parámetro s indica la cantidad máxima de dígitos almacenados a la derecha del punto decimal. s debe ser un valor de 0 a p. El valor predeterminado es 0	5-17 bytes
numeric (p, s)	Números de escala y precisión fijos. Permite números de $-10^{38} + 1$ a $10^{38} - 1$. El parámetro p indica el número total máximo de dígitos que se pueden almacenar (tanto a la izquierda como a la derecha del punto decimal). p debe ser un valor de 1 a 38. El valor predeterminado es 18. El parámetro s indica la cantidad máxima de dígitos almacenados a la derecha del punto decimal. s debe ser un valor de 0 a p. El valor predeterminado es 0	5-17 bytes
smallmoney	Datos monetarios de -214,748.3648 a 214,748.3647	4 bytes
money	Datos monetarios de -922,337,203,685,477.5808 a 922,337,203,685,477.5807	8 bytes

Tipos de datos de cadena

Tipo de datos	Descripción	Tamaño máximo	Almacenamiento
char (n)	Cadena de caracteres de ancho fijo	8,000 caracteres	Ancho definido
varchar (n)	Cadena de caracteres de ancho variable	8,000 caracteres	2 bytes + número de caracteres
varchar (max)	Cadena de caracteres de ancho variable	1,073,741,824 caracteres	2 bytes + número de caracteres
text	Cadena de caracteres de ancho variable	2 GB de datos de texto	4 bytes + número de caracteres
nchar	Cadena Unicode de ancho fijo	4,000 caracteres	Ancho definido x 2
nvarchar	Ancho de cadena Unicode	4,000 caracteres	
nvarchar (max)	Ancho de cadena Unicode	536,870,912 caracteres	
ntext	Ancho de cadena Unicode	2 GB de datos de texto	
binary (n)	Cadena binaria de ancho fijo	8,000 bytes	
varbinary	Cadena binaria de ancho variable	8,000 bytes	
varbinary (max)	Cadena binaria de ancho variable	2 GB	
image	Cadena binaria de ancho variable	2 GB	

3.1. Tipos de datos SQL

Tipos de datos de fecha

Tipo de datos	Descripción	Almacenamiento
float (n)	Datos del número de precisión flotante desde $-1.79E + 308$ a $1.79E + 308$. El parámetro n indica si el campo debe contener 4 u 8 bytes. float (24) contiene un campo de 4 bytes y float (53) contiene un campo de 8 bytes. El valor predeterminado de n es 53.	4 u 8 bytes
real	Datos numéricos de precisión flotante desde $-3.40E + 38$ a $3.40E + 38$	4 bytes
datetime	Del 1 de enero de 1753 al 31 de diciembre de 1999, con una precisión de 3,33 milisegundos	8 bytes
datetime2	Desde el 1 de enero de 0001 hasta el 31 de diciembre de 1999, con una precisión de 100 nanosegundos	6-8 bytes
smalldatetime	Del 1 de enero de 1900 al 6 de junio de 2079 con una precisión de 1 minuto	4 bytes
date	Almacenar una fecha solamente. Del 1 de enero de 0001 al 31 de diciembre de 9999	3 bytes
time	Almacenar un tiempo solo con una precisión de 100 nanosegundos	3-5 bytes
datetimeoffset	Lo mismo que datetime2 con la adición de un desplazamiento de zona horaria	8-10 bytes
timestamp	Almacena un número único que se actualiza cada vez que se crea o modifica una fila. El valor de la marca de tiempo se basa en un reloj interno y no corresponde a tiempo real. Cada tabla puede tener una sola variable de marca de tiempo	

3.2. Componentes de una Entidad en BD.

Componentes de una Base de Datos

El modelo de base de datos relacional se puede aplicar a múltiples estructuras de empresas comerciales, financieras y tecnológicas.

Los atributos son las características que queremos destacar de las entidades, se refieren a las personas, organizaciones, objetos o conceptos sobre los que nos interesa almacenar información.

Ese ID único de cada registro se conoce como *clave primaria (primary key o PK)*. Mientras que cada fila se puede vincular para crear una relación entre tablas diferentes, mediante una *clave externa (foreign key o FK)*. La clave externa se presenta como la clave primaria de otra tabla existente.

El modelo entidad relación de una base de datos relacional se compone de distintos elementos como la cardinalidad, las entidades, los atributos y las acciones.

3.2. Componentes de una Entidad en BD.

Una restricción de clave externa no tiene que estar vinculada solo a una restricción de clave principal en otra tabla. Las claves externas también se pueden definir para hacer referencia a las columnas de una restricción UNIQUE de otra tabla.

Si se especifica un valor distinto de NULL en la columna de una restricción FOREIGN KEY, el valor debe existir en la columna a la que se hace referencia. De lo contrario, se devuelve un mensaje de error de infracción de clave externa. Para asegurarse de que todos los valores de la restricción de clave externa compuesta se comprueben, especifique NOT NULL en todas las columnas que participan.

Las restricciones FOREIGN KEY solo pueden hacer referencia a las tablas de la misma base de datos en el mismo servidor. La integridad referencial entre bases de datos debe implementarse a través de desencadenadores. Para obtener más información, vea [CREATE TRIGGER..](#)

Las restricciones FOREIGN KEY pueden hacer referencia a otra columna de la misma tabla, lo que se conoce como autorreferencia.

3.2. Componentes de una Entidad en BD.

Una restricción FOREIGN KEY especificada en el nivel de columna solo puede incluir una columna de referencia. Esta columna debe tener el mismo tipo de datos que la columna en la que se define la restricción.

Una restricción FOREIGN KEY especificada en el nivel de tabla debe tener el mismo número de columnas de referencia que la lista de columnas de la restricción. El tipo de datos de cada columna de referencia debe ser también el mismo que el de la columna correspondiente de la lista de columnas.

Otras Restricciones

1.3. Modelo ER (entity-relationship)

Puntos para realizar el modelo Entidad-Relación de una BD

- a. Encontrar las entidades.
- b. Encontrar las relaciones.
- c. Encontrar los atributos y asociarlos a entidades y relaciones.
- d. Especificar las relaciones.
- e. Buscar los identificadores.
- f. Especificar los roles.
- g. Especificar las restricciones.
- h. Especificar las cardinalidades.
- i. Encontrar las claves.
- j. Dibujar el diagrama entidad-relación.
- k. Revisar el esquema conceptual.

3.3. Modelo ER (entity-relationship)

Reglas sobre la elección de Entidades

Para considerar la creación de una entidad, debe cumplir con las siguientes reglas:

1. Los nombres de las entidades deben representar el tipo de entidades (el conjunto de las ocurrencias) y no una ocurrencia de la misma.

Por ejemplo, un nombre apropiado para una entidad debe ser “Avión” y no “Boeing 727” o “Boeing 747”.

2. Debe tener múltiples ocurrencias. Una entidad con sólo una ocurrencia puede ser mejor representada como un atributo y no una entidad (no en todos los casos).

Una cosa u objeto debe poderse representar por una y solo una entidad. Las entidades deben ser mutuamente excluyentes en sus ocurrencias.

Debe poseer un identificador, un atributo que identifique únicamente a una ocurrencia de la entidad.

3.3. Modelo ER (entity-relationship)

Reglas sobre la elección de Entidades

Definición de una Entidad

(a:) Nombre: (Gp:) [PROFESOR](#)

(b:) Objeto: (Gp:) Almacenar la información relativa de los profesores de la organización.

(c:) Alcance: (Gp:) Se entiende como profesor a aquella [persona](#) que, contratada por la organización, imparte, al menos, un curso dentro de la misma.

(d:) Número de ejemplares: (Gp:) 10 profesores

(e:) Crecimiento previsto: (Gp:) 2 profesores / año

(f:) Observaciones: (Gp:) Los ejemplares dados de baja no serán eliminados de la base de datos; pasarán a tener una [marca](#) de eliminado y no serán visualizados desde la aplicación.

3.3. Modelo ER (entity-relationship)

Reglas sobre la elección de Entidades

¿Que son los Atributos? : Campos que describen de forma completa a una entidad y existen cuatro tipos de atributos:

Obligatorios: aquellos que deben tomar un [valor](#) y no se permite ningún ejemplar no tenga un valor determinado en el atributo.

Opcional: aquellos atributos que pueden tener [valores](#) o no tenerlo.

Derivado: aquellos atributos cuyo valor se obtiene a partir de [los valores](#) de otros atributos.

Claves: El modelo E-R exige que cada entidad tenga un identificador, se trata de un atributo o conjunto de atributos que identifican de forma única a cada uno de los ejemplares de la entidad.

De tal forma que ningún par de ejemplares de la entidad puedan tener el mismo valor en ese identificador.

3.3. Modelo ER (entity-relationship)

Reglas sobre la elección de Entidades

Reglas de los Atributos

- a. Los atributos deben ser univaluados, no pueden tener varios valores para una ocurrencia dada en un momento determinado.
- b. Pertenencia a la entidad. ¿Hace parte realmente de la entidad a la que asociamos?, si es a la BD suele utilizarse para nombrar a la cosa que es propiedad de una persona determinada (es decir, que tiene un dueño).
- c. Relevancia de un atributo depende del tipo del problema.

3.3. Modelo ER (entity-relationship)

Reglas sobre la elección de Entidades

Relaciones: Como su nombre indica es el elemento en el modelado que nos permite establecer una relación lógica entre entidades permitiendo modelar de forma mas completa el mundo real

James Martin: sugiere utilizar líneas para representar las relaciones incluyendo los nombres de las relaciones como etiqueta para esas líneas.

Los extremos de las líneas deben incluir algunos símbolos que señalen la cardinalidad de la relación. En esta notación, deben especificarse los dos nombres existentes para la relación.

(Gp:): PERSONA

(Gp:): [LIBRO](#)

(Gp:) Tiene

(Gp:) Pertenece a


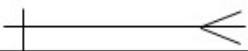
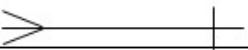

3.3. Modelo ER (entity-relationship)

Reglas sobre la elección de Entidades

Cardinalidad:

Expresan el número de atributos de una entidad con las que puede asociarse otro atributo dentro otra entidad mediante una relación.

- a. Importante: Cada relación debe contener un verbo .
- b. Cardinalidad Mínima. Indica el número mínimo de asociaciones en las que aparecerá cada ejemplar de la entidad (el valor que se anota es de cero o uno)
- c. Cardinalidad Máxima. Indica el número máximo de relaciones en las que puede aparecer cada ejemplar de la entidad (puede ser uno o muchos)

Cardinalidad	Se lee	Representación
1:1	Uno a uno	
1:M	Uno a muchos	
M:1	Muchos a uno	
M:M	Muchos a muchos	

3.3. Modelo ER (entity-relationship)

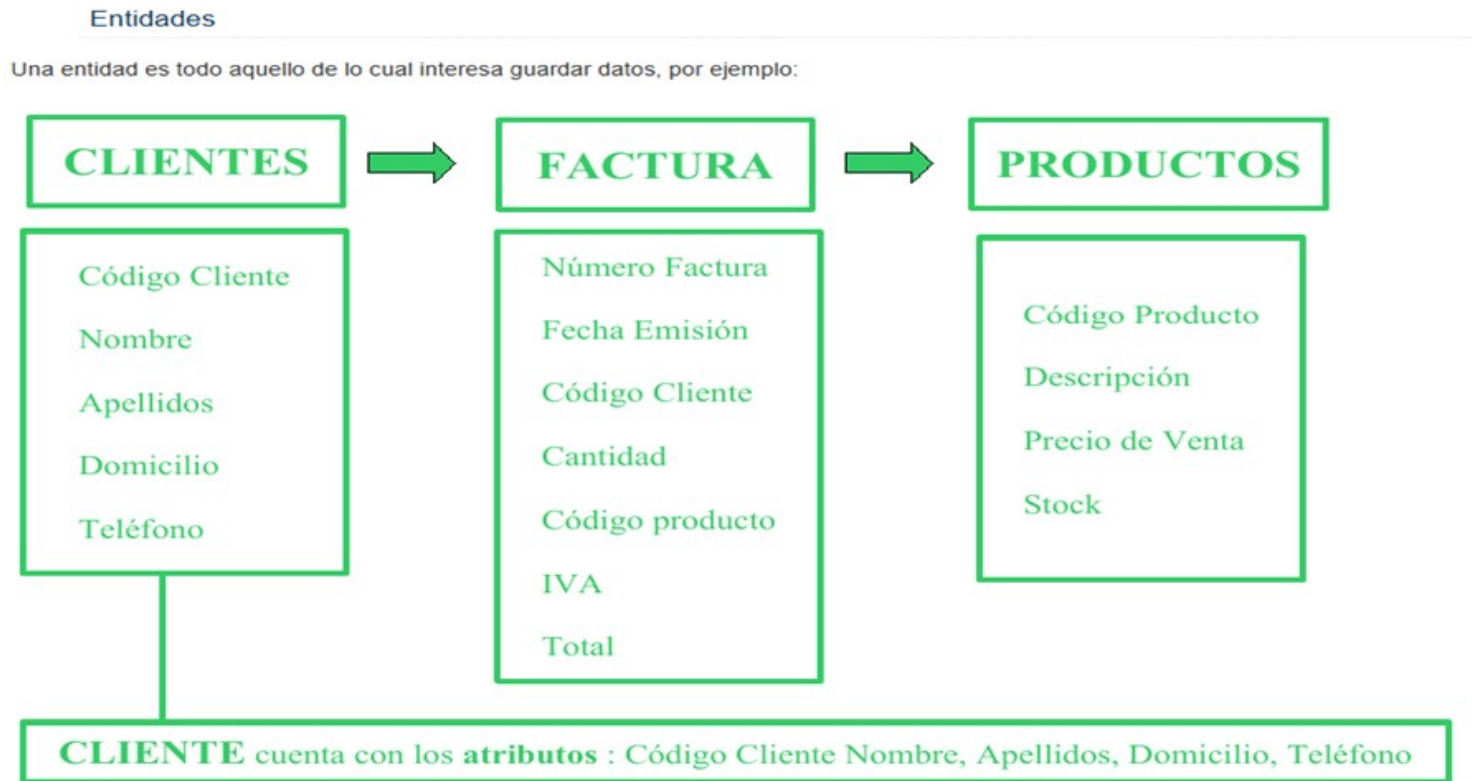
Reglas sobre la elección de Entidades

La cardinalidad en lo que respecta a el aspecto de instanciar una relación puede definirse de la siguiente manera:

- a. Una a Una: una entidad en A está asociada a lo sumo con una entidad en B, y una entidad en B está asociada a lo sumo con una entidad en A.
- b. Una a muchas: una entidad en A está asociada con un número cualquiera de entidades en B. Una entidad en B, sin embargo, puede estar asociada a lo sumo con una entidad en A.
- c. Muchas a Una: una entidad en A está asociada a lo sumo con una entidad en B, y una entidad en B, sin embargo, puede estar asociada con un número cualquiera de entidades en A.
- d. Muchas a muchas: una entidad en A está asociada con un número cualquiera de entidades en B, y una entidad en B está asociada con un número cualquiera de entidades en A.

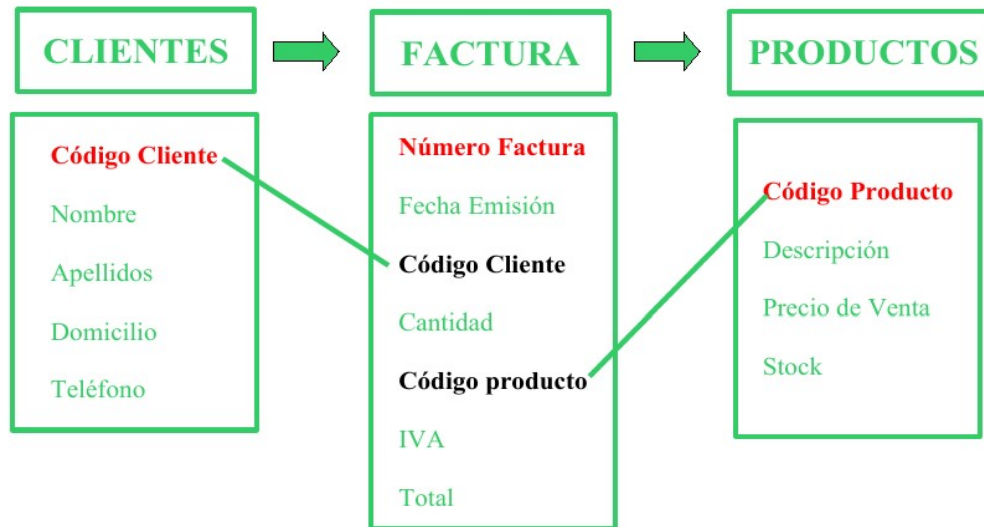
3.3. Modelo ER (entity-relationship)

Reglas sobre la elección de Entidades



3.3. Modelo ER (entity-relationship)

Reglas sobre la elección de Entidades



CLAVES PRIMARIAS

- **Código Cliente** es la clave primaria de **CLIENTES**. A cada cliente se le asocia un código y a cada código le corresponde un cliente.
- **Número Factura** es clave primaria de **FACTURAS**.
- **Código Producto** es clave primaria de **PRODUCTOS**.

CLAVES FORÁNEAS

- En **FACTURAS**, son claves foráneas **Código Cliente** y **Código Producto**. **CLIENTES** se relaciona con **FACTURAS** a través del Código Cliente que figura en ambas tablas y con **PRODUCTOS** mediante el Código Producto.

1.3. Modelo ER (entity-relationship)

ELIMINAR EN CASCADA: Cuando creamos una clave foránea utilizando esta opción, elimina las filas de referencia en la tabla secundaria cuando la fila referenciada se elimina en la tabla primaria que tiene una clave primaria. Es importante indicar que esta opción no cuenta con un deshacer por lo cual su aplicación debe ser cuidadosa

ACTUALIZAR CASCADA: Cuando creamos una clave externa utilizando ACTUALIZAR CASCADA, las filas de referencia se van a actualizar en la tabla secundaria cuando la fila referenciada se actualiza en la tabla principal que tiene una clave primaria. Su utilización es adecuada y conveniente toda vez que nos ayuda a realizar actualizaciones masivas de las claves o i llaves sin tener que ir tabla a atabla

Estas actividades han sido desarrolladas y explicadas en los laboratorios del curso:


1. Creación de la regla ELIMINAR y ACTUALIZAR CASCADA en una clave externa usando el estudio de administración de SQL Server
2. Crear la regla ELIMINAR y ACTUALIZAR CASCADA en una clave foránea utilizando el script T-SQL
3. Se lanzará en una tabla con una clave externa en cascada ELIMINAR o ACTUALIZAR

3.4. Integridad de una Entidad de BD

El modelo entidad relación de una base de datos relacional se compone de distintos elementos como la cardinalidad, las entidades, los atributos y las acciones.

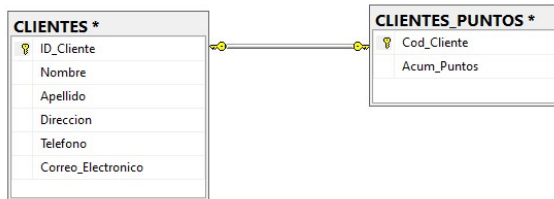
Recordemos que las **entidades** hacen referencia a las tablas de la base datos y se representan en rectángulos. Por ejemplo, una empresa de venta de indumentaria puede tener la Tabla Clientes, la Tabla Compras y la Tabla Artículos.

Recordemos que las **entidades** hacen referencia a las tablas de la base datos y se representan en rectángulos. Por ejemplo, una empresa de venta de indumentaria puede tener la Tabla Clientes, la Tabla Compras y la Tabla Artículos.

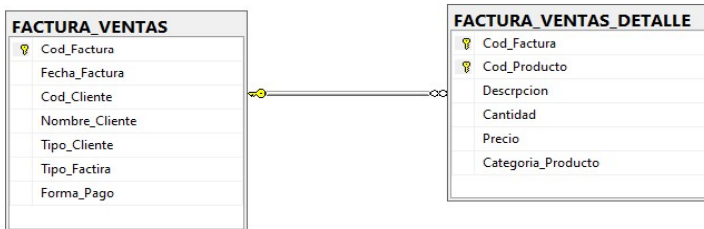


3.4. Integridad de una Entidad de BD

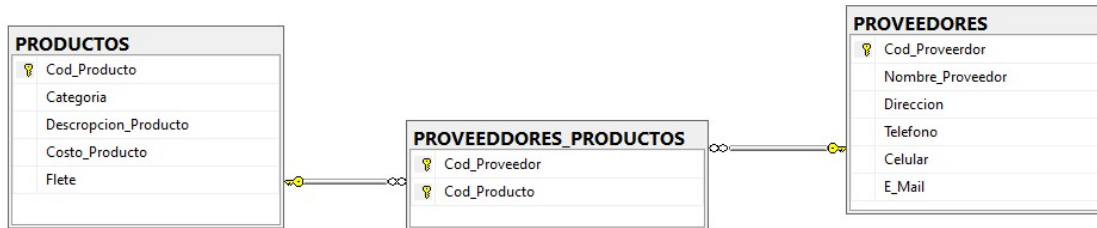
Uno a Uno (1:1)



Uno a muchos (1:M)



Muchos a muchos (M:M)



3.4. Modelo Físico (Tablas)

Tabla a

DNI	Nombre	Altura	Edad	Ciudad
50234561	Juan Gómez	1,85	35	Madrid
13254673	Eduarne Montero	1,60	30	Toledo
46576290	Luis Urrutia	1,75	46	Madrid
38941882	Juan Gómez	1, 71	55	Valencia

Tabla b

ID	Nombre	Altura	Edad	Ciudad
001	Juan Gómez	1,85	35	Madrid
002	Eduarne Montero	1,60	30	Toledo
003	Luis Urrutia	1,75	46	Madrid
004	Juan Gómez	1, 71	55	Valencia

La tabla a) contiene un atributo único (DNI).

La tabla b) no contiene un atributo único entre sus datos, pero se añade el campo ID con un código arbitrario que puede ser empleado como clave.

El nombre en este caso no sirve como atributo único, ya que hay dos personas en la tabla con el mismo nombre