**FORMATO PARA EL DESARROLLO DE COMPONENTE FORMATIVO**

|  |  |
| --- | --- |
| PROGRAMA DE FORMACIÓN | Análisis y desarrollo de *software* |

|  |  |  |  |
| --- | --- | --- | --- |
| COMPETENCIA | 220501095. Diseñar la solución de software de acuerdo con procedimientos y requisitos técnicos. | RESULTADOS DE APRENDIZAJE | 220501095-02. Estructurar el modelo de datos del software de acuerdo con las  especificaciones del análisis. |

|  |  |
| --- | --- |
| NÚMERO DEL COMPONENTE FORMATIVO | 5 |
| NOMBRE DEL COMPONENTE FORMATIVO | Conceptos generales de bases de datos |
| BREVE DESCRIPCIÓN | En la sociedad de la información es imprescindible el empleo de tecnología, técnicas y procedimientos que, a lo largo de las últimas décadas, han consolidado un marco conceptual importante y necesario en todo proceso de administración de datos, por eso, este componente acerca estos conceptos de manera estructurada, para avanzar en los procesos de gestión de la información |
| PALABRAS CLAVE | Bases de datos, gestión, entidad, relación |

|  |  |
| --- | --- |
| ÁREA OCUPACIONAL | 2 - CIENCIAS NATURALES, APLICADAS Y RELACIONADAS |
| IDIOMA | Español |

1. **TABLA DE CONTENIDOS:**

# **Conceptos generales de base de datos**

## 1.1 Tipos de datos y restricción de no nulidad

## 1.2 Tipos de bases de datos

## 1.3 Clasificación de bases de datos

## 1.4 Sistema de gestión de bases de datos

# **Modelo entidad relación**

## 2.1 Relaciones entre entidades

2.1.1 Principio de unicidad

2.1.2 Claves

2.1.3 Claves candidata, superclave y clave primaria

## 2.2 Relaciones de uno a muchos

## 2.3 Relación de muchos a muchos

## 2.4 Relaciones de uno a uno

# **Normalización**

## 3.1 Dependencias funcionales

3.1.1 Definición formal dependencia funcional

## 3.2 Diseño relacional

## 3.3 Reglas de integridad

3.3.1 Regla de borrado

3.3.2 Regla de edición

## 3.4 Lenguajes de los sistemas administradores de bases de datos

3.4.1 Sentencias de definición de datos (DDL)

3.4.2 Sentencias de manejo de datos (DML)

3.4.3 Sentencias de control (DCL)

3.4.4 Sentencias de control de transacciones (TCL)

# **Sistema gestor de base de datos**

1. **INTRODUCCIÓN**

Una base de datos se puede percibir como un “almacén” de información que se define y se crea una sola vez para guardar grandes cantidades de datos de forma organizada (o estructurada), con el fin de poder encontrarla y utilizarla fácilmente.

Teniendo en cuenta la anterior información, en el presente componente formativo, se presentará el concepto y características de las bases de datos actuales, tomando en cuenta que cada base de datos es diseñada para cumplir con los requisitos de información de una organización o una empresa.

Estos diseños están concebidos para emplear Sistemas de Gestión de Bases de Datos (en adelante SGDB), que es un sistema o servicio informático que permite a las personas definir, crear, dar soporte y mantenimiento a las bases de datos, controlando el acceso de forma segura.

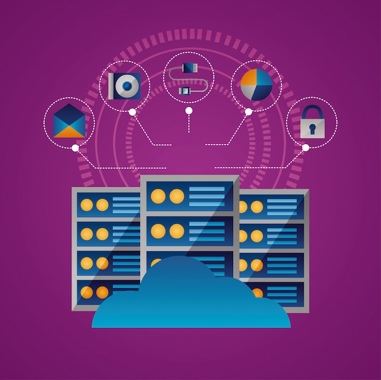
Bienvenido a este proceso de aprendizaje.

1. **DESARROLLO DE CONTENIDOS:**

# **Conceptos generales de base de datos**

Como concepto general de bases de datos podemos encontrar:

“El conjunto de datos organizado de tal modo que permita obtener con rapidez diversos tipos de información” (RAE, 2001).



De acuerdo con esta definición, una hoja de cálculo de Excel puede considerarse una base de datos, o un conjunto de archivos debidamente organizados, o la lista de nombres y teléfonos que está en nuestros *smartphones*. En principio, es correcto llamarle base de datos a estos ejemplos, sin embargo, en el contexto de desarrollo de *software*, se referirá a ese conjunto de información que puede ser almacenada en grandes cantidades de forma organizada y es gestionada a través de un **Sistema de Gestión de Bases de Datos** (SGBD).

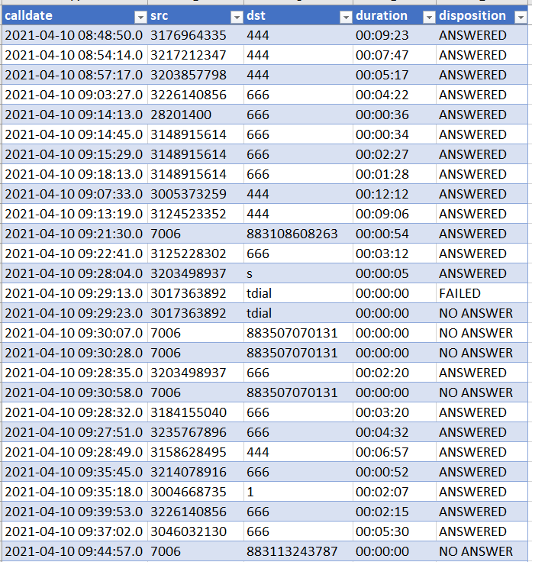
Para organizar y definir la información de forma sistemática, las bases de datos también deben poder almacenar una descripción precisa de los datos que contiene, conocida como **metadatos,** a los que se le relaciona el tipo de información que es conceptualmente guardada (es decir, si se agrega una explicación de la naturaleza del dato en la empresa), se da origen a lo que conoce como **diccionario de datos o catálogo de datos.**

**Ejemplo 1 - Determinar los metadatos del siguiente problema**

Una empresa tiene una base de datos de llamadas telefónicas que entran y salen de su planta telefónica, donde se relacionan los números de teléfonos del llamante y la extensión telefónica que recibe o hace la llamada, la fecha y la hora de la llamada, si fue o no atendida, y la duración de la misma como se muestra en la siguiente figura.

**Figura 1**

Base de datos de llamadas telefónicas



La figura muestra una tabla de datos que donde en cada columna se agrupa un tipo de dato particular, y cada fila es el conjunto de datos que se llama registro y comparten una relación (la relación es que cada fila corresponde a una única llamada telefónica); a esta estructura se le llama bidimensional, aquí filas y columnas son las dos dimensiones.

Los metadatos de la figura anterior serían la definición de cada tipo de dato como se lista a continuación.

**calldate**: tiene la estructura YYYY-MM-DD HH:MI:SS.Z, donde:

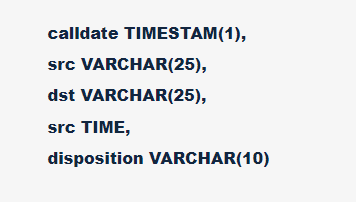
* YYYY sería el año un dato numérico entero de cuatro dígitos.
* MM el mes un dato numérico entero de dos dígitos entre 01 y 12.
* DD un dato numérico entero de dos dígitos entre 01 y 31.
* HH un dato numérico de dos dígitos entre 00 y 23.
* MI y SS un dato numérico de dos dígitos entre 00 y 59
* Z un dato numérico de un digito (para representar milésimas de segundo).

Como este es el conjunto de datos de la primera columna, puede que use un término que describa lo anterior con una sola palabra que es TIMESTAMP, de esta forma esa primera columna puede quedar correctamente descrita como:

**calldate TIMESTAMP(1)** Donde 1 es el número de dígitos de milisegundos (Z).

src y dst: empleado para fuente de la llamada o llamante (src) y destino de la llamada (dst), puede ser un número telefónico o de extensión telefónica. Algunos tienen el código de país seguido del signo más, ejemplo: 57+3155008002. Por lo tanto, se puede describir como una cadena de texto que no superará los 25 caracteres, pero puede tener menos de 25 caracteres, así el metadato iría descrito:

* **src VARCHAR(25)** De esta forma decimos que es una cadena de caracteres.
* **dst VARCHAR(25)** Que varía en longitud de 0 a 25 caracteres.
* **duration** Tiene la estructura HH:MI:SS y como se vio antes, representa el tiempo en horas, minutos y segundos: **src TIME TIME** se usa para esta estructura de dato en particular.
* **disposition** Indica si la llamada es atendida ANSWER, no atendida NO ANSWER, o si falló la FAIL. Se puede representar con una cadena de 10 caracteres.



Note que cada uno de ellos fue separado por una coma, para que se diferencie de los demás.

**Ejemplo 2**

Conociendo el objetivo de la base de datos y los metadatos de la figura anterior, crear un diccionario de datos o un catálogo de datos.

Solución:

**Tabla 1**

Diccionario de datos

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre | Diccionario de datos base de llamadas telefónicas | | |
| Creación | 27/02/2021 | | |
| Descripción | Registro de las llamadas telefónicas de la PBX de la empresa STD LTDA. | | |
| Campo | **Tipo dato** | **Tamaño** | **Descripción** |
| calldate | TIMESTAMP | 1 | Representa el momento exacto en que entra o sale la llamada. |
| src | VARCHAR | 25 | Fuente (*source*) de la llamada, número de quien hace la llamada telefónica. |
| dst | VARCHAR | 25 | Número del destino de la llamada telefónica. |
| duration | TIME | 0 | Duración de la llamada telefónica. |
| disposition | VARCHAR | 10 | Estado final en que se considera quedó la llamada. |

Como se puede notar en la tabla anterior, un diccionario de datos aporta información de la estructura de los datos y el uso que se da a cada dato en la empresa, organización o sistema de información, pero no sin antes darle un nombre a todo el conjunto de datos (primera fila), una fecha (segunda fila) y una descripción (tercera fila) del conjunto de los datos. En la tabla se puede identificar la independencia que existe entre la lógica de los datos (representado por la columna DESCRIPCIÓN del dato) y el almacenamiento físico (estructura de almacenamiento representado por las columnas TIPO DE DATO y TAMAÑO).

## **1.1 Tipos de datos y restricción de no nulidad**

Existen muchos tipos de datos y varían según el SGDB, porque cada sistema gestor define sus propios tipos de datos, aunque existen las equivalencias notables y otras no tan notables. A continuación, se presenta una tabla que relaciona el tipo de datos según los SGDB más comunes del mercado, como ORACLE, PostgeSQL, MySQL y SQLServer.

**Tabla 2**

Tipos de datos según bases de datos

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tipo de dato | ORACLE | PostgreSQL | MySQL | SQLServer |
| Cadena caracteres | VARCHAR2 | CHARACTER VARYING | VARCHAR | VARCHAR |
| Cadena texto | TEXT | TEXT | TEXT | NTEXT |
| Entero pequeño | SMALLINT | SMALLINT | SMALLINT | SMALLINT |
| Entero | INTEGER | INTEGER | INT | INT |
| Fecha | DATE | DATE | DATE | DATE |
| Fecha y hora | DATE | TIMESTAMP WITH TIME ZONE | DATETIME | DATETIME2 |
| Hora | DATE | TIME | TIME | TIME |
| Entero con decimales | FLOAT | REAL | FLOAT | FLOAT |

Según el tipo de sistema gestor de bases de datos, cada uno define sus tipos de datos que, de alguna forma, son muy parecidos entre ellos, o existen equivalencias; las diferencias son pocas y están relacionadas con aspectos técnicos del almacenamiento. Se pueden consultar más detalles de cada uno de ellos en los sitios oficiales de cada motor de base de datos:

|  |  |
| --- | --- |
| Nombre | Enlace de acceso |
| ORACLE | <https://docs.oracle.com/cd/E11882_01/server.112/e41085/sqlqr06002.htm#SQLQR959> |
| PostgreSQL | <https://www.postgresql.org/docs/13/datatype.html#DATATYPE-TABLE> |
| MySQL | <https://dev.mysql.com/doc/refman/8.0/en/data-types.html> |
| SQLServer | <https://docs.microsoft.com/en-us/sql/t-sql/data-types/data-types-transact-sql?view=sql-server-ver15> |

Existen varios tipos de restricciones, por ahora se verá la restricción de longitud, por ejemplo:

**src VARCHAR(25)**

En él se define que el src, es una columna que no puede exceder los 25 caracteres. De la misma forma, existe una forma de definir que es obligatoria: registrar un dato en esa columna y se llama **restricción de no nulidad**, es decir, que ninguna fila puede tener valor nulo. Así, en el ejemplo de la base de datos de llamadas telefónicas, si se define que todo y cada uno de los datos son obligatorios (usando las palabras **NOT NULL**), la metadata que define la estructura de los datos, quedaría así:

* **calldate TIMESTAM(1) NOT NULL,**
* **src VARCHAR(25)NOT NULL,**
* **dst VARCHAR(25)NOT NULL,**
* **src TIME NOT NULL,**
* **disposition VARCHAR(10) NOT NULL**

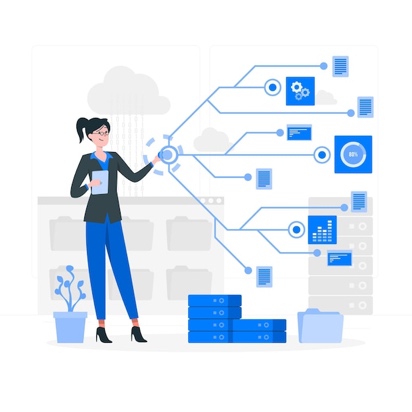
Algunos sistemas, para abreviar en la decisión de no nulidad, acostumbra usar el acrónimo NN para representar NOT NULL.

## 

## **1.2 Tipos de bases de datos**

Las bases de datos han evolucionado a lo largo del tiempo, usando diferentes tipos de modelos o enfoques, se ha comprobado que algunos de ellos tienen mejor desempeño en distintos tipos de aplicación. La estructura en la que se configuran los datos al interior de las bases de datos presenta una forma de clasificación (Kyocera, 2021):

|  |
| --- |
| CF8\_1\_2\_Tipos\_bases\_datos |

Cada tipo de bases de datos tiene un ámbito de aplicación en el que su desempeño es mejor. Por ejemplo, las bases de datos jerárquicas tienen un mejor desempeño en operaciones de consulta de datos puntuales, mientras que las de datos relacionales son precisas para garantizar la calidad de los datos y la no repetición de los mismos, aunque no muy optimizadas para hacer consultas, y, finalmente, las multidimensionales, son muy útiles para análisis estadístico de datos históricos de volúmenes inmensos de datos. Aparte del tipo de datos, existe otro tipo de clasificaciones ya no según la estructura en que se almacenan los datos, sino de acuerdo con la naturaleza de los datos.

## 

## **1.3 Clasificación de bases de datos**

Los datos se pueden clasificar según la naturaleza de los datos que se almacenan en la base de datos. La siguiente tabla muestra una compilación de estas clasificaciones.

**Tabla 3**

Clasificación

|  |  |  |
| --- | --- | --- |
| Criterio de clasificación | Clasificación | Descripción |
| Según la variabilidad de los datos | **Bases de datos estáticas** | * Son bases de datos cuyos datos son históricos, es decir, ya no se pueden modificar, se usan comúnmente para estudiar el comportamiento de los datos a través del tiempo. * Generalmente, a este tipo de bases de datos se les llama bodega de datos. * Muchas de estas bases de datos se almacenan en cubos para su análisis, y se les conoce como OLAP es el acrónimo en inglés de Procesamiento Analítico en Línea (*On-Line Analytical Processing*). |
| **Bases de datos dinámicas** | * Estos datos se almacenan y pueden ser modificados, agregados, borrados y consultados en cualquier momento, por ejemplo: un sistema de facturación. * A estos sistemas se les denomina transaccionales, porque cada operación de guardar, borrar o editar, se configura como una transacción. OLTP es la sigla en inglés de Procesamiento de Transacciones en Línea (*On-Line Transaction Processing*). |
| Según el contenido | **Bases de datos documentales** | * Permiten la indexación a texto completo, y en líneas generales realizar búsquedas más potentes. |
| **Base de datos deductivos** | * Un sistema de base de datos deductivos, es un sistema de base de datos, pero con la diferencia que permite hacer deducciones a través de inferencias. Se basa principalmente en reglas y hechos que son almacenados en la base de datos. También las bases de datos deductivas son llamadas base de datos lógica, a raíz que se basan en lógica matemática. |

Pueden existir otras clasificaciones orientadas al ámbito de uso, sin embargo, se presentan las más comunes en el ejercicio del desarrollo de *software.* Las bases de datos relacionales es de obligatorio dominio en cualquiera de los casos, debido a que los conceptos de estas son reutilizables en casi todos los otros tipos.

## **1.4 Sistema de gestión de bases de datos**

Un sistema de gestión de la base de datos es un programa de computador que permite definir, crear y mantener los datos de una base de datos, controlando el acceso:

* Permiten la definición de la base de datos usando un lenguaje de definición de datos.
* Permiten la inserción, actualización, eliminación y consulta de datos usando un lenguaje de manejo de datos.
* Proporcionan un acceso controlado a la base de datos (con autenticación, roles, niveles de acceso).
* Concurrencia (varios usuarios a la vez accediendo o manipulando los datos) y multitarea.
* Algunos SGDB permiten administrar el catálogo de datos.

Los SGBD son una herramienta muy útil, sin embargo, se podría decir que los SGBD son un poco más complejos, porque los usuarios ven más datos y sus relaciones, de los que realmente se necesitan.

# **Modelo entidad relación**



Las bases de datos relacionales, inicialmente no son fáciles de modelar, por lo que se requiere de cierto nivel de abstracción, es decir, se va ocultando la complejidad del diseño de la base de datos, primero analizando el requerimiento desde un punto de vista conceptual, entendido como las necesidades de la empresa o del sistema de información, luego se va analizando desde el punto de vista de la lógica de los datos y, finalmente, se analiza físicamente cómo se almacenarán los datos. A continuación, se presentan algunos conceptos:

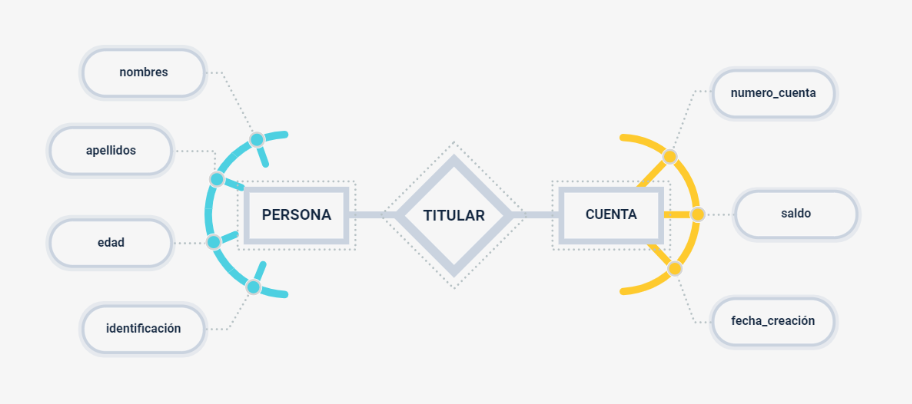
|  |
| --- |
| CF8\_2\_Conceptos |

Se presentará más claro en un diagrama, pero antes se deben considerar los siguientes elementos:

1. Rectángulos, que representan conjuntos de entidades.
2. Elipses, que representan atributos.
3. Rombos, que representan relaciones entre conjuntos de entidades.
4. Líneas, que unen los atributos con los conjuntos de entidades y los conjuntos de entidades con las relaciones.

**Figura 2**

Entidad relación persona cuenta bancaria



Para una base de datos de un sistema bancario consistente en clientes (es decir personas) y sus cuentas bancarias, en la figura anterior se muestra el diagrama Entidad Relación. En ells se indica que hay dos entidades que son persona y cuenta, con los atributos descritos anteriormente para cada una. El diagrama también representa la relación titular entre persona y cuenta.

Se debe tener en cuenta que el modelo relacional convierte las entidades en tablas, por lo tanto, se puede poner como ejemplo que las entidades personas se conviertan en tablas como muestra la siguiente figura.

**Figura 3**

Tablas de una entidad relación

**º**

Las tablas de la figura anterior muestran cada uno de los conjuntos de registros, la “tabla persona” muestra 6 registros y la “tabla cuenta” muestra 9 de ellas; cada uno de estos registros representa un objeto, es decir, de la tabla persona hay 6 (objetos) o personas. A la agrupación de los datos de cada registro se le llama TUPLA y, en términos sencillos, una TUPLA de una base de datos, es el conjunto de todos los atributos (columnas) de una fila.

## 

## **2.1 Relaciones entre entidades**

Como se observa en las figuras anteriores, una entidad se puede representar con una tabla, sin embargo, la relación titular aún no se sabe cómo representarla y para ello se debe definir algunas propiedades de los atributos.

**2.1.1 Principio de unicidad**

En una tabla nunca deben existir TUPLAS repetidas, es decir, no debe haber más de una fila con exactamente los mismos valores en cada columna que otra fila de la misma tabla. En otras palabras, cada fila debe diferir al menos en un elemento (columna) en relación con las demás.

En una base de datos relacional, las relaciones permiten evitar datos redundantes. En el ejemplo anterior, una persona puede tener varias cuentas bancarias, pero no es deseable que, por cada una de las cuentas bancarias asociadas a la misma persona, se deban repetir los datos de la misma persona, sino más bien, se debe encontrar un mecanismo para relacionar la información de las dos tablas y para ello se introducen algunos conceptos útiles.

**2.1.2 Claves**

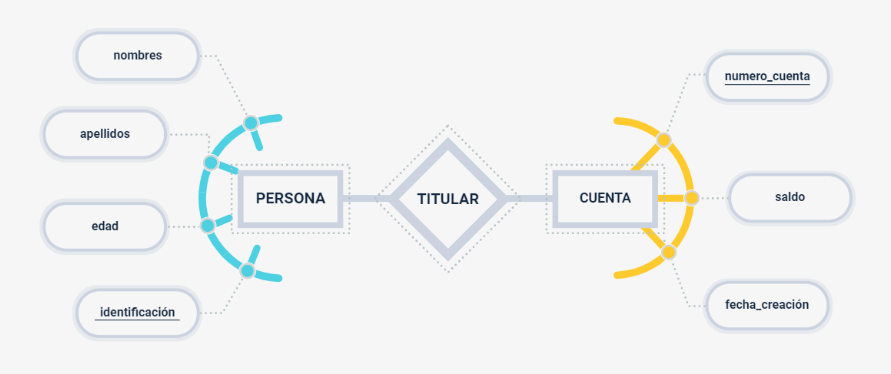
Dado que en una tabla no hay filas repetidas, se pueden distinguir unas de otras, es decir, se pueden identificar de manera única (por el principio de unidad). Por lo tanto, la manera de identificarlas inequívocamente es según los valores de los atributos. De esta forma, se conoce como claves a los atributos que sirven para identificar un registro de forma inequívoca.

**Ejemplo:**

En la tabla persona de la figura 3, es sabido por todos que no existen dos personas con el mismo número de documento de identidad, por lo tanto, se puede emplear el atributo identificación como clave de la tabla persona. Así mismo, no deben existir dos cuentas bancarias con el mismo número de cuenta, entonces, el atributo numero\_cuenta se puede emplear como clave de la tabla cuenta. Esto se puede presentar en un diagrama de entidad relación cuando a los atributos que componen una clave, se les marca con subrayado el nombre del atributo tal como se muestra en la siguiente figura.

**Figura 4**

Claves de las entidades persona y cuenta



Nótese como los atributos **identificacion y numero\_cuenta** tiene subrayada la palabra identificadora del atributo, representado así que ese atributo es la clave (que hace que se cumpla el criterio de unicidad) de su entidad.

**2.1.3 Claves candidata, superclave y clave primaria**

Supongamos el caso que el banco es internacional, que es un banco moderno y solo diseñado para transacciones vía internet o virtuales, por lo tanto, el correo del cliente (persona) es necesario y obligatorio.

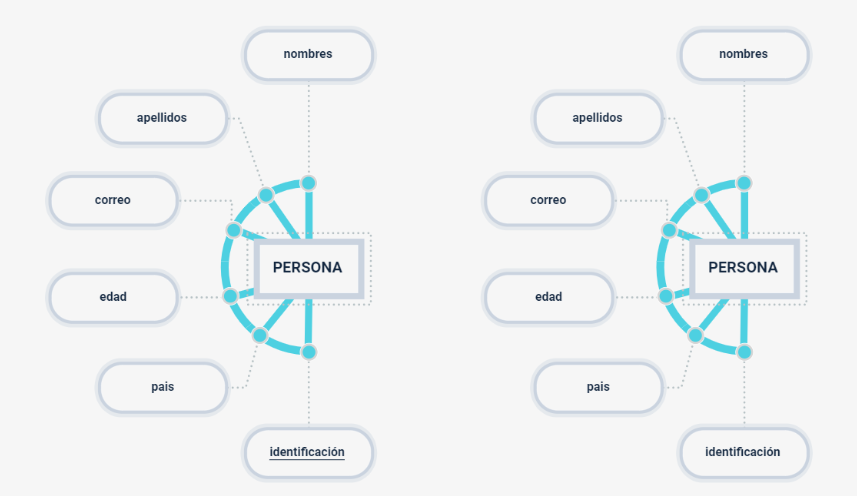
En este caso, ocurren un par de asuntos a analizar: al ser internacional el banco, puede existir un cliente en un país con el mismo número de identificación de otro cliente en otro país, por lo tanto, el atributo **identificación** ya no puede ser una clave de la entidad, sin embargo, es importante y obligatorio saber a qué país corresponde cada cliente, el conjunto de datos **identificación y país** y si ambos cumplen con el criterio de unidad, porque no existen dos personas en el mismo país con el mismo número de identificación.

**Clave compuesta:** si se escogiera el conjunto de atributos **identificación y país** como clave de la entidad, se está haciendo una clave compuesta, porque es un conjunto de más de un atributo el que se usa para este fin.

El otro asunto a considerar es que se puede emplear el correo electrónico como clave, porque no existen dos usuarios con el mismo **correo** electrónico, esto es válido si se definió como regla del banco que cada cliente tenga su correo electrónico diferente de los demás. Estos dos casos se representan en la siguiente figura.

**Figura 5**

Claves candidata

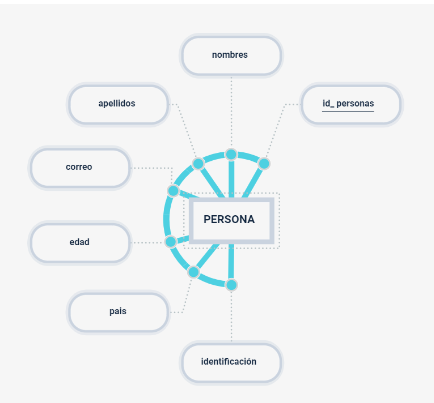


**Clave candidata:** los dos caminos propuestos en la figura anterior son válidos, y se denominan claves candidatas porque con ambas se puede lograr el principio de unicidad, y es potestad de cada diseñador o analista de sistemas, determinar cuál de las dos emplear.

Sin embargo, existe otro tipo de solución y es asignarle un atributo identificador a la entidad persona que se encargue exclusivamente de identificar inequívocamente cada tupla o fila, a este atributo se le puede llamar id\_persona que sería un número que el programa o *software* para el que se está desarrollando la base, le asignará a cada cliente cada vez que se registren por primera vez, los datos de una persona (generalmente una secuencia). De esta forma, se tendrían 3 claves candidatas; la siguiente figura presenta cómo sería la entidad y la tabla que representa.

**Figura 6**

Claves candidatas secuencia



Este último enfoque es el más empleado y recomendado por los diseñadores o analistas de software, pues reduce, significativamente, la complejidad del diseño y facilita la relación con otras tablas o entidades. Por lo tanto, se usará cada vez que se tenga oportunidad de hacerlo.

**Superclave:** es un conjunto de uno o más atributos que, tomados juntos, permiten identificar de forma única una entidad en el conjunto de entidades. Por ejemplo, el atributo id\_persona del conjunto de entidades persona, es suficiente para distinguir una persona cliente del banco de las otras. Así, id\_persona es una superclave. Análogamente, la combinación de identificación y país es una superclave del conjunto de entidades persona. El atributo correo de persona es otra superclave, pero el atributo nombres no lo es, porque varias personas podrían tener el mismo nombre.

**Clave primaria:** se emplea el término clave primaria para nombrar una clave candidata que es elegida por el diseñador o analista de la base de datos como clave principal para identificar las tuplas dentro de una entidad. Una clave (primaria, candidata y superclave) es una propiedad del conjunto de entidades, más que de las entidades individuales. Dos tuplas individuales en el conjunto, no pueden tener el mismo valor en sus atributos clave primaria al mismo tiempo. La designación de una clave primaria es en sí una restricción que modela una característica del mundo real.

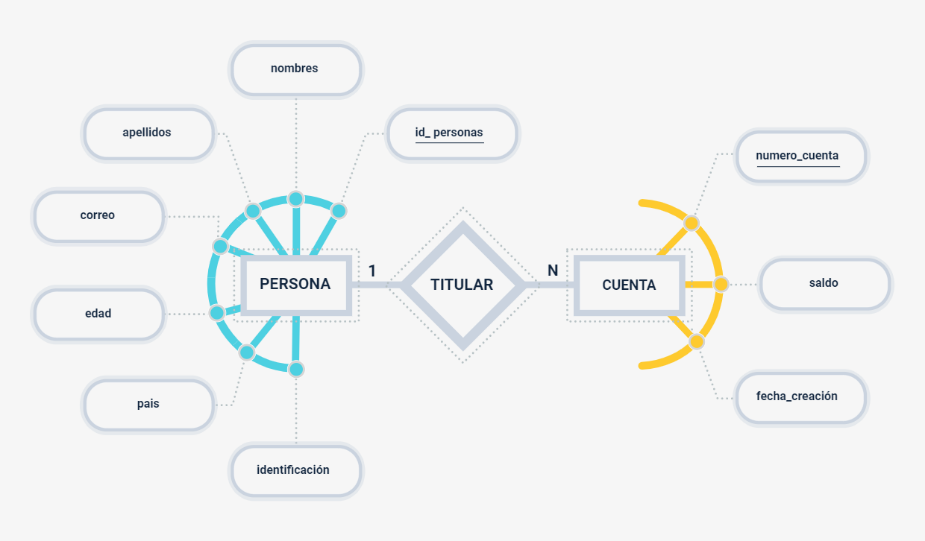
## **2.2 Relaciones de uno a muchos**

Una relación de uno a varios o de uno a muchos, es el tipo de relación más empleada. En este tipo de relación, una fila de la tabla A puede tener muchas filas coincidentes en la tabla B. Pero una fila de la tabla B solo puede tener una fila coincidente en la tabla A. Por ejemplo, las tablas persona y cuenta tienen una relación de uno a muchos. Es decir, cada persona puede tener más de una cuenta, pero, por ejemplo, cada título proviene de una sola editorial.

En un diagrama de entidad relación esto se representa con lo que se denomina multiplicidad, como se presenta en la siguiente figura.

**Figura 7**

Relaciones de uno a muchos

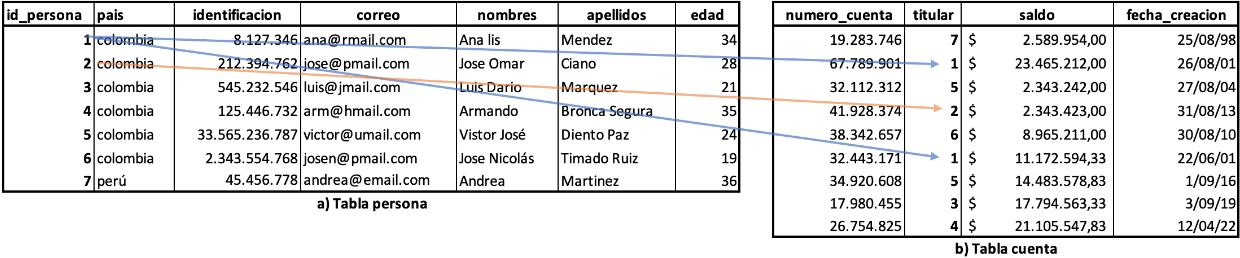
****

A esta representación se le denomina multiplicidad **1:N** y se lee de la siguiente forma: **UNA persona** es titular de **N cuentas** bancarias, de esta forma una persona puede tener N (que puede ser cero, uno o más de una) cuentas bancarias. Se puede leer en forma inversa que el resultado es el mismo: muchas o varias cuentas bancarias pueden tener un mismo titular.

Cuando una relación de es uno a muchos y se mapea un modelo de base de datos relacional, la relación **titular** (rombo) se termina convirtiendo una columna de la tabla de los muchos.

**Figura 8**

Tablas de un modelo relacional 1:N

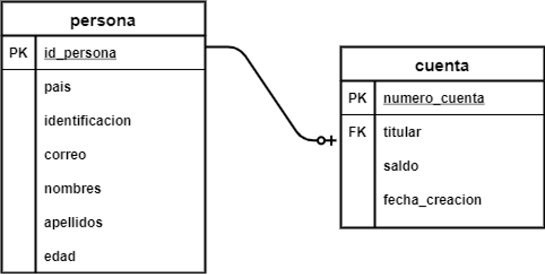
****

Como se puede ver en la figura anterior, Ana Lis Méndez, es titular de dos cuentas bancarias (67.789.901 y 32.443.171), y todos los datos de la columna titular en la tabla cuenta se conoce como **LLAVE FORÁNEA** de la tabla persona.

**REGLA DE MAPEO 1:N de Modelo entidad-relación a modelo relacional:** una relación de uno a muchos (1:N), se transforma en una columna en la tabla que tiene la multiplicidad de los muchos (cuenta). Y todos los datos de esta columna existen como llave primaria en la tabla que tiene la multiplicidad uno (tabla persona).

**Figura 9**

Diagrama relacional 1:N

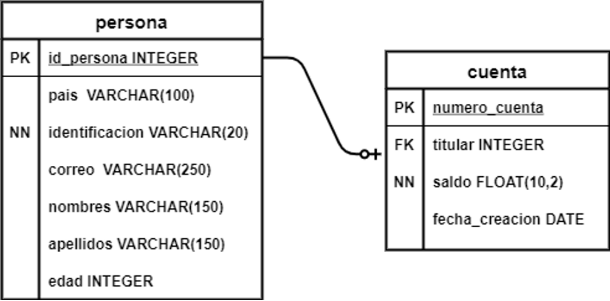


En la figura se puede ver el nombre de las entidades (persona y cuenta), los atributos haciendo especial énfasis en los que representan la llave primaria con el indicador PK (del inglés *Primary Key*), y también se observa la relación titular (columna de la tabla), indicando que es una clave o llave foránea de la tabla persona, lo que se indica con el indicador FK (del inglés *Foreign Key*).

Algunos diagramas relacionales agregan metadatos para hacer más claro el modelo al diseñador, así como la naturaleza de los datos del modelo.

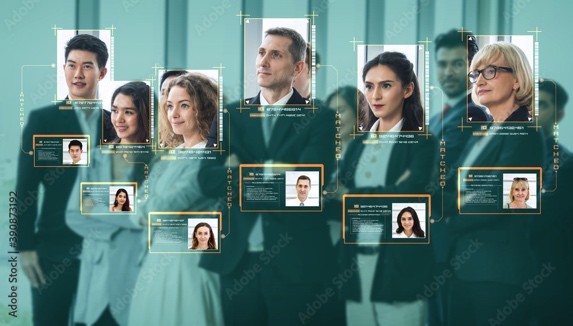
**Figura 10**

Diagrama relacional 1:N con metadatos

****

**Conclusión:** la relación de uno a muchos se termina convirtiendo en una columna en la tabla que tiene la multiplicidad de los muchos.

## **2.3 Relación de muchos a muchos**



ÇEn una relación de muchos a muchos, una fila de la tabla A puede tener relación con muchas filas en la tabla B y también, en sentido contrario, una fila de la tabla B con respecto de varias filas de la tabla A. Esta relación se crea definiendo una tercera tabla denominada tabla de relación. Esta nueva tabla tiene una clave principal de la tabla compuesta de las claves foráneas de la tabla A y de la B.

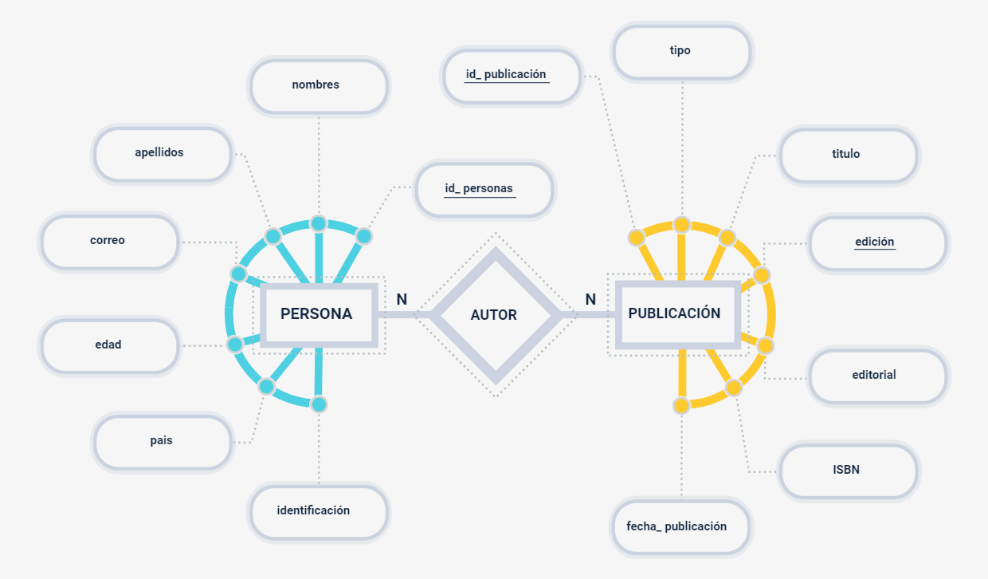
**Ejemplo:**

Se necesita almacenar la información de unas referencias bibliográficas o publicaciones de libros, tesis, artículos, que escriben los estudiantes de una universidad, y como se sabe, un libro, tesis, artículo o cualquier publicación puede ser elaborada por más de una persona, y a su vez, una persona (docente, estudiante o investigador) puede tener más de una publicación de su autoría.

Como se observa en la siguiente figura, la entidad publicación tiene una llave primaria llamada identificación, y existe una relaciona llamada **autor**, y representa que una persona puede ser autor de una o muchas publicaciones, y que una publicación puede ser de autoría de una o muchas personas, la multiplicidad en este caso es de muchos a muchos y se representa como N:N.

**Figura 11**

Diagrama de entidad de publicaciones bibliográficas

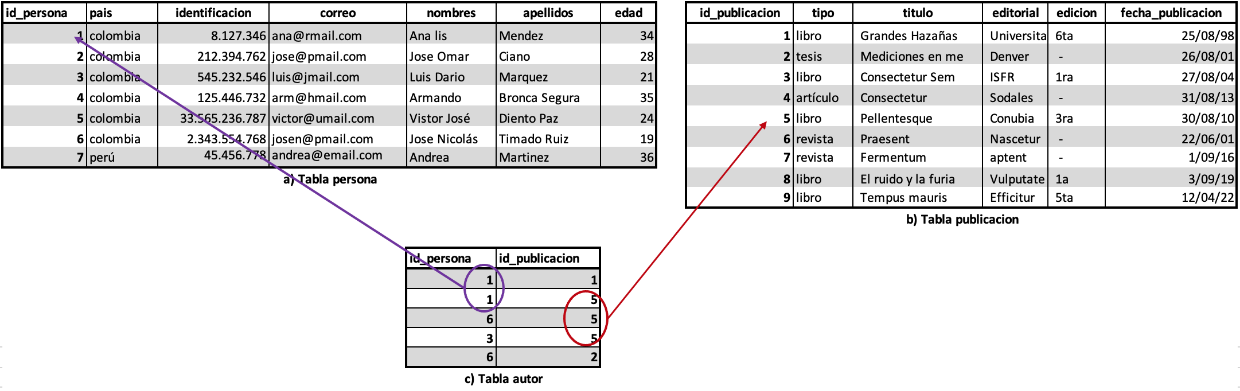


**REGLA DE MAPEO N:N** **de modelo entidad-relación a modelo relacional:** en una relación de muchos a muchos (N:N) se transforma en una tabla, cuya llave primaria está compuesta por las llaves primarias de las otras dos tablas.

Como se puede ver en la siguiente figura, la “tabla autor”, dice que la persona Ana Lis, es autora de los libros de título Grandes hazañas y Pellentesque, también que el libro Pellentesque es escrito además de Ana Lis por Luis Darío y Jose Nicolás. Lo anterior haciendo referencia a las llaves primarias de las tablas a) persona y b) publicación.

**Figura 12**

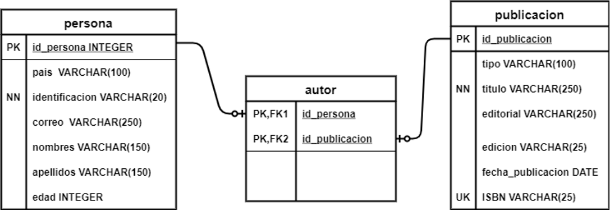
Tablas de relación N:N

****

La representación de la relación N:N con un diagrama relacional se muestra en la figura a continuación, donde se usan los prefijos **FK**1 y **FK**2, para indicar llave foránea uno y dos, para las entidades persona y publicación 2, respectivamente. También se puede ver el prefijo **UK** (*Unique Key*) en la tabla publicación, en la columna ISBN, el cual significa que en toda la tabla publicación no puede existir más de una fila con el mismo ISBN, es decir, nunca ninguna publicación tendrá el mismo ISBN que otra.

**Figura 13**

Diagrama relacional N:N

****

## **2.4 Relaciones de uno a uno**

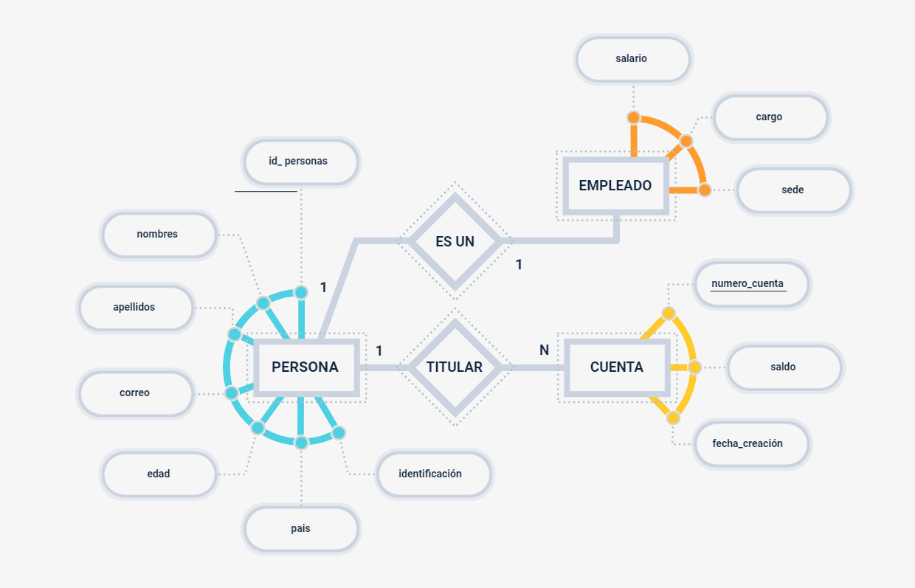
En una relación uno a uno, una fila de la tabla A solo puede tener una fila coincidente en la tabla B y viceversa. Se crea, entonces, una relación uno a uno si ambas columnas relacionadas son claves principales o tienen restricciones únicas. Este tipo de relación no es común, porque la mayoría de la información que está relacionada de esta manera estaría en una tabla. Se puede usar una relación uno a uno por alguno de estos motivos:

|  |
| --- |
| CF8\_2\_4\_Uno\_a\_uno |

Volviendo al ejemplo del banco, esta vez se supone que la empresa necesita información sobre sus empleados lo que, a la vez, pueden ser clientes del banco.

**Figura 14**

Diagrama entidad relación persona-empleado

****

**REGLA DE MAPEO 1:1 de modelo entidad-relación a modelo relacional**: en el momento de convertir la relación o con multiplicidad 1:1 a tablas, la relación uno a uno se convierte en una columna de la tabla empleado con restricción de unicidad (*Unique Key* o *Primary Key*), como se muestra en el siguiente diagrama relacional.

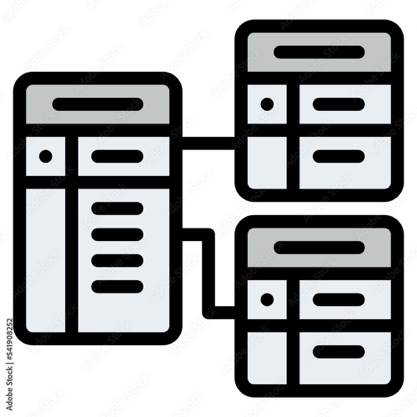
Como se puede inferir de la siguiente figura, se crea una tabla donde se relacionan los datos de las personas que son empleados del banco, a través de una columna que se llama **id\_persona**, pero es común entre algunos diseñadores ponerle el nombre que los relacione con la tabla a la que pertenece, es decir, **id\_empleado**, pero esto queda al criterio del diseñador.

**Figura 15**

Diagrama relacional 1:1

****

# **Normalización**

La normalización es el procedimiento mediante el cual se aplican las reglas de mapeo o conversión de un modelo entidad-relación a un modelo relacional, es decir, a tablas y relaciones. Los principios rectores de este proceso son: no redundancia de datos (que no se repitan los datos) y que la dependencia de datos sea coherente (una separación lógica de datos en tablas).

Los datos redundantes desperdician espacio en disco y crean problemas de mantenimiento. Por ejemplo, en la figura 15 del tema anterior, se ve que los datos comunes de los clientes y los bancos están en la tabla persona, mientras que los datos que pertenecen exclusivamente a los empleados están en otra tabla que se relaciona con la de personas.

Ahora, ¿qué es una dependencia incoherente?

Aunque resulta intuitivo para un usuario buscar en la tabla de persona el correo de un cliente en particular, es posible que no tenga sentido buscar en esa columna el salario del empleado que se relaciona con esos datos de la tabla persona, porque el salario del empleado está relacionado con la tabla empleado o depende de él. Las dependencias incoherentes pueden dificultar el acceso a los datos, porque la ruta para encontrar los datos puede faltar o estar dañada.

A continuación, lo invitamos a ver el siguiente video sobre normalización.



La siguiente tabla nos presenta las tres formas normales y sus principios.

**Tabla 4**

Formas normales

|  |  |
| --- | --- |
| Forma normal | Principios |
| Primera forma normal | * Eliminar grupos de repetición en tablas individuales. * Crear una tabla independiente para cada conjunto de datos relacionados. * Identificar cada conjunto de datos relacionados con una clave primaria. |
| Segunda forma normal | * Crear tablas independientes para conjuntos de valores que se aplican a varios registros. * Relacionar estas tablas con una clave foránea. |
| Tercera forma normal | * Eliminar los campos que no dependen de la clave. |

Hay una cuarta forma normal, también llamada forma normal de ***Boyce Codd* (BCNF)**, y una quinta forma normal, pero rara vez se consideran en un diseño práctico, solo agregan complejidad al sistema y no aportan un valor funcional que aporte a la solución del problema, por esta razón no será tratada.

## 

## **3.1 Dependencias funcionales**

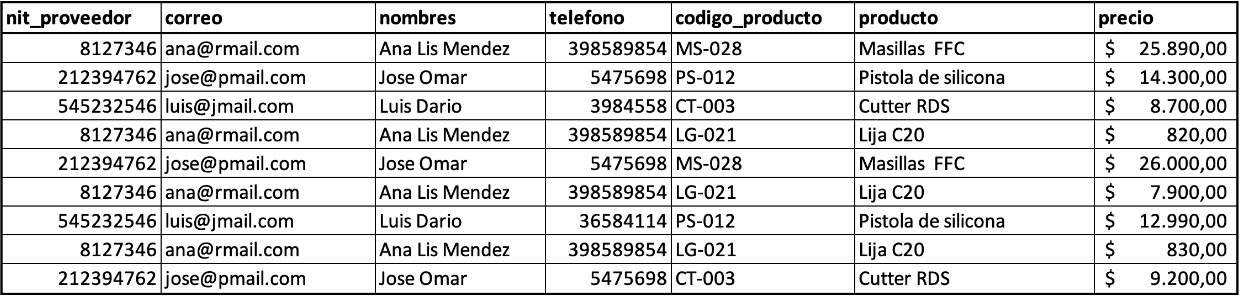
Una dependencia funcional es un tipo de restricción que termina construyendo una generalización del concepto de clave, como se estudió en el modelo E-R y en el modelo relacional. Pero no es tan fácil localizar las dependencias, debido a que necesitan de un análisis de los atributos (columnas) o, con más precisión, de las interrelaciones entre atributos y, frecuentemente, la intuición no es suficiente a la hora de encontrar y clasificar todas las dependencias. Aunque existe una teoría matemática para realizar este análisis, un ejemplo sencillo puede enseñar cómo analizar estas dependencias de manera intuitiva, tal como se presentará a continuación.

**Ejemplo**

Un cliente pide que se desarrolle un *software* para llevar el inventario de una ferretería, los productos, los proveedores y el precio al que cada proveedor vende cada producto, para lo cual suministra la siguiente información, y se deben identificar las dependencias funcionales para poder realizar un modelo relacional.

**Figura 16**

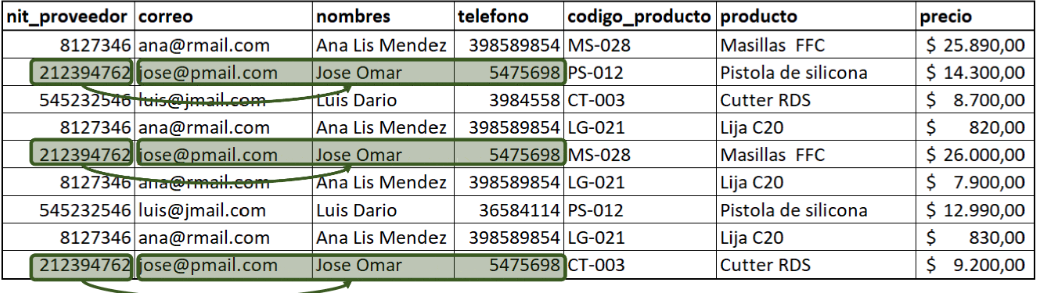
Proveedores, productos y precios



El principio de no repetición de datos, indica que se deben identificar todos los datos que se repiten y se puede ver en la siguiente figura, tenido en cuenta los colores de sombreado como se muestra. Es de notar que el nit\_proveedor es igual en todos los registros que tienen el mismo valor de correo, nombres y teléfono.

**Figura 17**

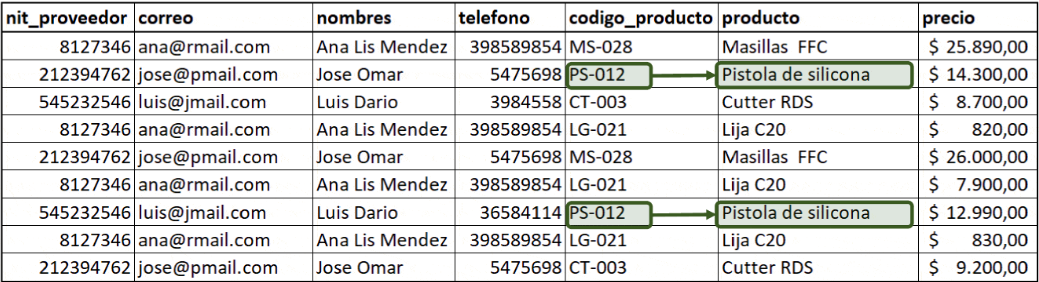
Dependencia de nit\_proveedor con datos del proveedor



Ahora, en la siguiente figura, la columna código\_producto es la misma para la columna producto, pero difiere del precio que cada proveedor ofrece del mismo.

**Figura 18**

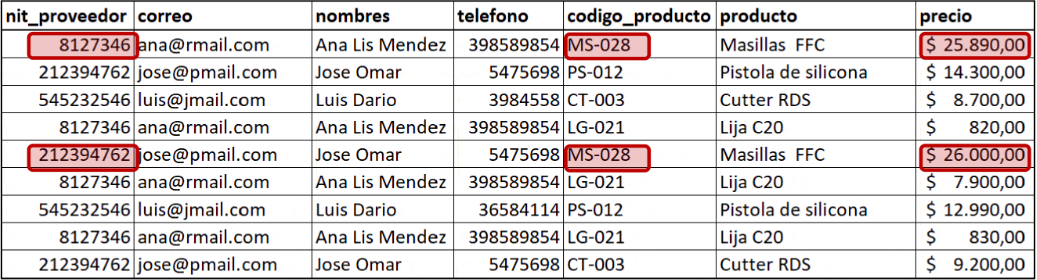
Dependencia de código\_producto\_ con datos del producto



La dependencia **código\_producto** no se cumple para el precio del producto, por lo tanto, dice que para cada proveedor y producto, debe existir un precio distinto, como se presenta en la siguiente figura.

**Figura 10**

Dependencia de nit\_proveedor, código\_producto\_ con precio del producto



En la siguiente tabla, se observan las dependencias detectadas.

**Tabla 5**

Dependencias funcionales

|  |  |
| --- | --- |
| Atributo | Es dependencia funcional de: |
| nit\_proveedor | * Correo * Nombres * Teléfono |
| codigo\_producto | * Producto |
| nit\_proveedor, codigo\_producto | * Precio |
| Referencia Tabla - Norma APA | |

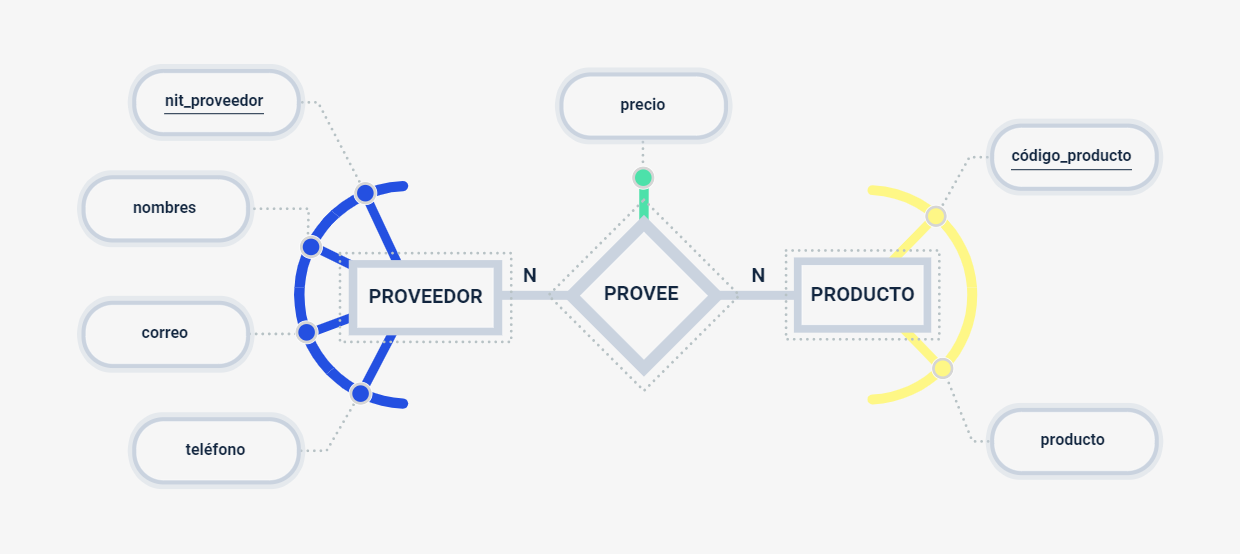
**3.1.1 Definición formal dependencia funcional**

Se dice que un atributo X de una relación, “depende funcionalmente” de otro atributo o conjunto de atributos Y de la relación, si a todo valor (o valores del conjunto) Y le corresponde siempre el mismo valor de X.

Las dependencias funcionales de la tabla anterior se pueden representar en modelo entidad relación, teniendo en cuenta que un proveedor puede suministrar muchos productos y, a su vez, un producto puede ser proveedor por más de un proveedor, y que por cada producto que es proveído por un proveedor, existe un precio.

**Figura 20**

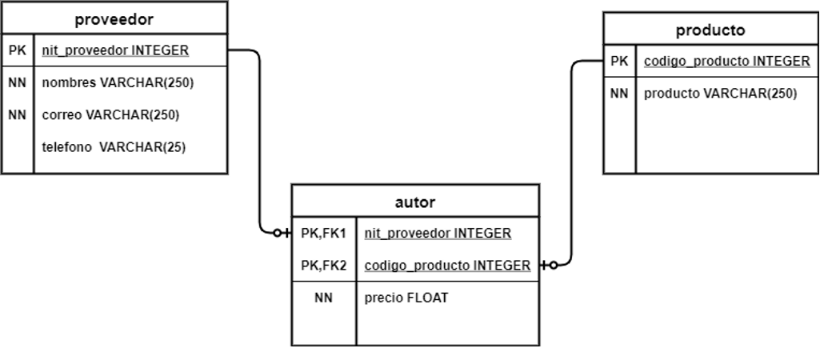
Diagrama entidad relación dependencia funcional

****

Como se puede notar, en la figura anterior, la relación provee, cuando es mapeado a un diagrama relacional, se genera una tabla intermedia provee, que debe tener un atributo adicional “precio”.

**Figura 21**

Diagrama relacional dependencia funcional

****

## **3.2 Diseño relacional**

Hasta este momento, se han examinado problemas concretos de las formas normales muy básicas y de la normalización. Ahora, se estudiará el modo en que se encaja la normalización en un proceso general de diseño de bases de datos y se puede hacer de una de las siguientes formas:

|  |
| --- |
| CF8\_3\_2\_Formas |

En el mundo práctico y en la medida que se tiene más experticia, las opciones 2 y 3 son las más empleadas.

A continuación, se muestra un ejemplo completo sobre cómo abordar un problema de almacenamiento de datos y transformarlo en un modelo relacional, recogiendo los conceptos vistos hasta ahora, pasando de un diagrama entidad relación a un modelo relacional, que es en todo caso el objetivo general de un diseño de base de datos.

**Problema:**

Se necesita gestionar una biblioteca escolar y nuestro cliente quiere tener ciertas herramientas a su disposición para controlar publicaciones tales como libros, revistas, folletos y también llevar un registro de los estudiantes y préstamos. Adicionalmente, se necesita un control de los ejemplares de cada libro o publicación, su ubicación y su estado, registrar la retirada de libros por préstamo y restitución de ejemplares, para esto necesita información sobre editoriales datos como dirección, teléfono y nombre de la editorial.

De cada publicación se pueden tener varios ejemplares, por ejemplo: del libro Nacho lee, la biblioteca tiene 5 ejemplares, también de cada publicación se debe conocer, su autor o autores, el tema que trata la publicación (geografía, física, matemática, etc.) y la editorial. Cada libro tiene un título, idioma, formato (digital web, impreso, digital CD), código ISBN y fecha de publicación.

Cada ejemplar tiene un código de barras, cuando un ejemplar ya está muy descargado se debe poder dar de baja; es decir, que ya se puede botar y no usarlo más.

Tanto el estudiante, como el autor de un libro tienen los siguientes datos: nombres, apellidos, correo. Adicional, un estudiante tiene los datos de documento de identidad, teléfono, dirección, grado escolar.

De cada préstamo interesa saber qué ejemplar se prestó, fecha en la que se prestó, la fecha en que se devolvió y alguna nota o comentario.

**Figura 22**

Análisis de relacionar autor

****

En este diagrama se presenta una posible solución:

La relación entre persona y estudiante corresponde al caso estudiado en las relaciones de uno a uno (entre persona y empleado del banco), de la figura Diagrama relacional 1:1.

El tema se vuelve complejo, porque se debe analizar si aplica lo mismo para autor.

Al mapear estas entidades a diagrama relacional, resultan 3 tablas (estudiante, persona y autor), pero autor solo tendría una única columna y al no tener atributos adicionales relacionados con el autor, es prácticamente lo mismo que tener la tabla persona, por lo tanto, no es conveniente dejarlo de esta forma y más bien relacionar la publicación directamente con la entidad persona.

Se debe estudiar detenidamente el diagrama entidad relación por varios minutos, asegúrese que comprende el porqué de cada elemento y su multiplicidad.

**Para recordar:**

Una relación de muchos a muchos (N:N) se convierte en una tabla cuya llave primaria es computar por las llaves foráneas de las otras dos tablas. Una relación de uno a muchos (1:N) se convierte en una columna (llave foránea) de la tabla que tiene los muchos, y finalmente una relación de uno a uno (1:1) se convierte una columna en cualquier (llave foránea) de las dos tablas.

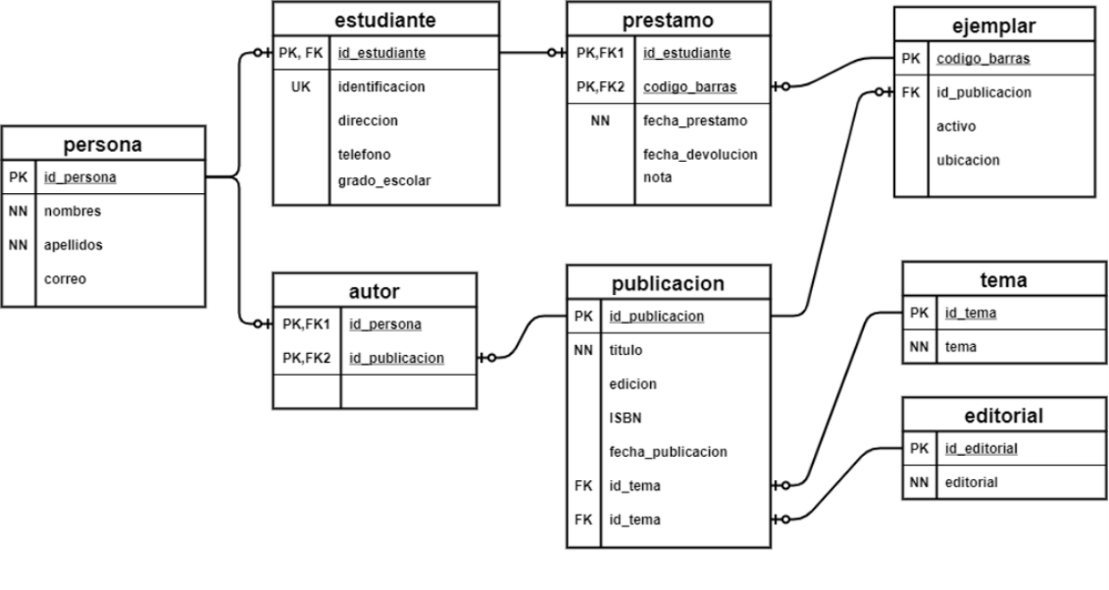
**Modelo entidad biblioteca**

Lo invitamos a descargar el PDF Modelo entidad relación biblioteca, para conocer cómo queda el diagrama con el ejercicio trabajado.

A continuación, se presenta el modelo racional.

**Figura 23**

Modelo relacional

****

Hasta ahora se ha visto cómo distribuir el almacenamiento de los datos en tablas, de manera que no se permita la duplicidad de datos y procurando una estructura lógica, que presente la información del mundo real. Pero aún falta analizar el problema desde el punto de vista que permita garantizar que los datos que se van a insertar, son correctos, para lo cual es importante conocer las restricciones de integridad que puede tener una base de datos.

## **3.3 Reglas de integridad**

Cuando se define un atributo, sin saber, se está escogiendo para él un dominio, es decir, un conjunto de valores que puede tener. Por ejemplo: en MySQL, cuando se elige un atributo de tipo entero (INT), los posibles valores que puede tener esa columna van desde -2147483648 hasta 2147483647, es decir, valores positivos o negativos, de manera que si se quieren solo valores positivos, se deben definir como entero sin signo (UNSIGNED INT), por lo tanto, los valores van desde 0 hasta 4294967295.

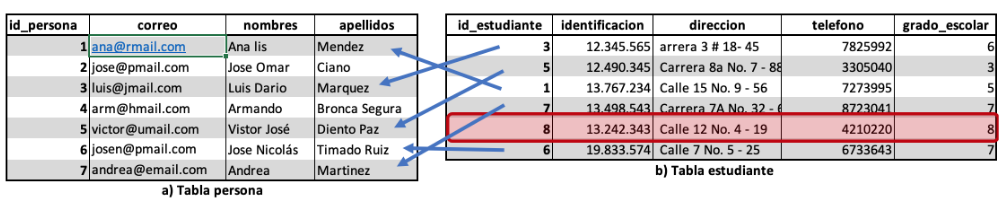
La calidad de los datos garantiza que los datos almacenados en la base de datos cumplan con los estándares y requisitos de la organización. En otras palabras, asegura el mantenimiento de la integridad de los datos en una base de datos. Al hacerlo, aplica un conjunto de reglas a un conjunto de datos completo o específico y se almacena en la base de datos de destino, por lo consiguiente, para garantizar la integridad se debe tener en cuenta:

|  |
| --- |
| CF8\_3\_3\_Tener\_en\_centa |

En la siguiente figura, por cada estudiante (tabla estudiante) existe uno y solo un elemento en la tabla persona, de tal forma que un estudiante es la conjunción de ambas tablas por medio de la llave foránea (que a la vez es llave primaria) id\_estudiante, pero ¿qué tal que no existiera? Eso significa que la base de datos es inconsistente o que tiene datos basura en la tabla estudiante. Afortunadamente los SGDB permiten definir reglas que impidan que esto ocurra en las tablas, es decir, que no permita registrar un estudiante con id\_estudiante con un valor que no existe en la columna id\_persona de la tabla persona.

**Figura 24**

Error de integridad referencial

****

**Pregunta de análisis de datos**

Ahora, hay que preguntarse lo siguiente: ¿qué pasaría si se borra el registro de la tabla persona que tiene id\_persona con valor tres (3), sufrirá algún problema los datos de la tabla estudiante?, o ¿qué pasaría si se cambia el valor de tres (3) en la columna id\_persona, por otro valor como por ejemplo nueve (9)?, antes de continuar, hay que tomarse unos minutos para pensarlo.

**Respuesta**

La respuesta a las cuestiones planteadas es que en cualquiera de las operaciones de borrado o actualización del dato se incurrirá en una violación de la integridad referencial de la tabla estudiante, porque no habrá manera de conocer los datos complementarios del estudiante que se encuentran en la tabla persona.

Para evitar estos problemas de inconsistencia, los sistemas gestores de bases de datos permiten definir unas reglas cuando se borra o actualiza un dato que es referencia foránea de otra tabla.

**3.3.1 Regla de borrado**

Define el comportamiento a la pregunta: ¿qué ocurre si se intenta borrar la fila referenciada por la clave foránea de otra tabla? Para lo cual, el SGBD implementa una de las siguientes opciones:

|  |
| --- |
| CF8\_3\_3\_Borrado |

**3.3.2 Regla de edición**

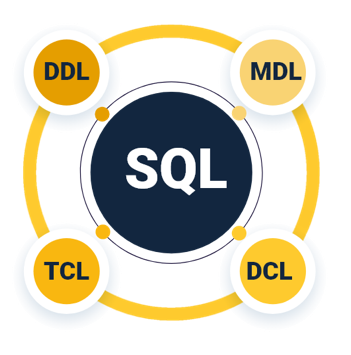
Define el comportamiento a la pregunta: ¿qué sucede si se intenta modificar el valor de la clave primaria de la fila referenciada por la clave foránea por otra tabla? Para lo cual, el SGBD implementa una de las siguientes opciones:

|  |
| --- |
| CF8\_3\_3\_Edicion |

De esta forma, una vez definida la estructura del modelo de datos, se debe determinar el comportamiento de estas operaciones, lo cual generalmente responde a requerimiento del sistema de información.

## 

## **3.4 Lenguajes de los sistemas administradores de bases de datos**

****

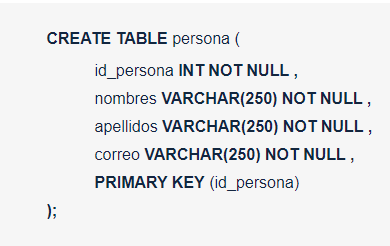
Los sistemas gestores de bases de datos, emplean un lenguaje que se denomina SQL que corresponde al nombre en inglés (*Structured Query Language*). Generalmente, cuando un SGBD relacional implementa el lenguaje SQL, todas las acciones, y operaciones se llevan a cabo en el sistema, mediante sentencias de este lenguaje. Dentro de SQL hay varios tipos de sentencias que se agrupan en cuatro conjuntos.

**3.4.1 Sentencias de definición de datos (DDL)**

Son las sentencias que permiten crear tablas, alterar su definición y eliminarlas. En una base de datos relacional existen otros tipos de objetos además de las tablas, como las vistas, los índices y los disparadores. Las sentencias para crear, alterar y eliminar vistas e índices también pertenecen a este conjunto de sentencia y se les denomina DDL del inglés *Data Definition Language*.

**Ejemplo DDL**

Para crear la tabla persona de la figura Modelo relacional, quedaría de la siguiente forma:



Nota: cada atributo se define en una línea separado por una coma, se agregan restricciones de no nulidad (NOT NULL) al final la sentencia termina con punto y coma. Ahora, así se borraría esa misma tabla:



**3.4.2 Sentencias de manejo de datos (DML)**

Estas sentencias que permiten insertar datos en las tablas, consultarlos, editarlos y borrarlos, se denomina DML del inglés *Data Manipulation Language.*

**Ejemplo DML**

A continuación, se muestra cómo se inserta una fila a la tabla persona, luego cómo se actualiza la fila insertada y posteriormente cómo se borra:

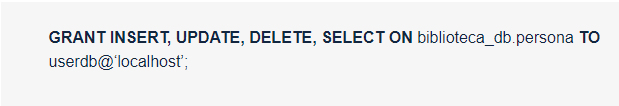


**3.4.3 Sentencias de control (DCL)**

Son las sentencias empleadas por los administradores de la base de datos para realizar tareas como por ejemplo, crear usuarios y concederles o revocar los privilegios. Se usa el término DCL del inglés *Data Control Language.*

**Ejemplo DCL**

En el ejemplo se le da permisos a un usuario llamado userdb de la base de datos de nombre biblioteca\_db para que pueda insertar, actualizar, borrar y consultar datos de la tabla persona desde el mismo computador donde está instalada la base de datos (localhost).



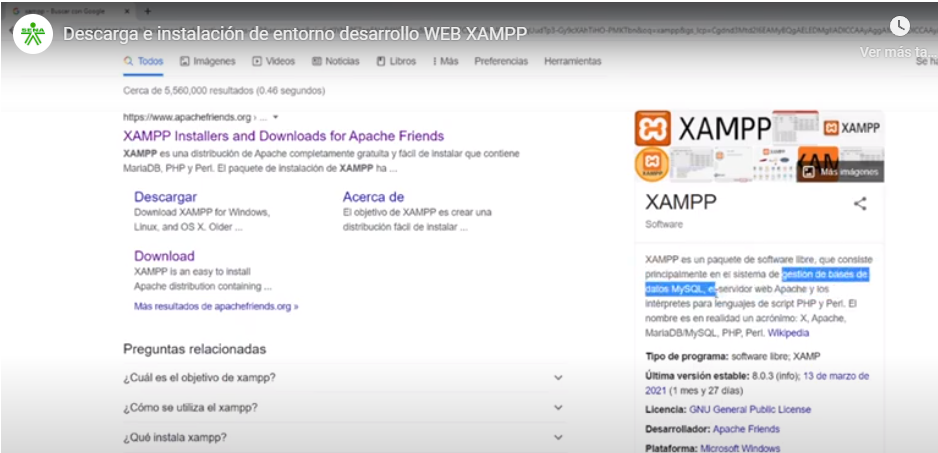
**3.4.4 Sentencias de control de transacciones (TCL)**

Un pequeño grupo de sentencias que permiten procesar en bloque operaciones DML, garantizando que se efectúen todas y cada una de las operaciones o ninguna. **TCL** del inglés *Transaction Control Language.*

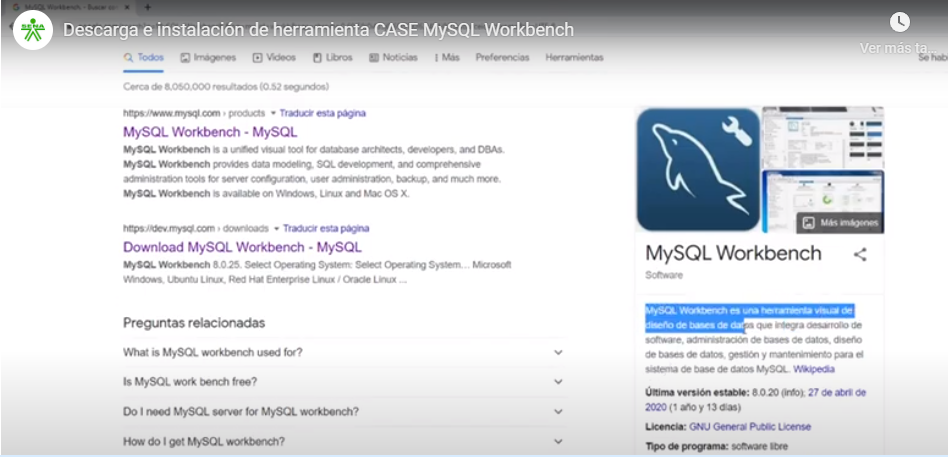
# **Sistema gestor de base de datos**

En el mercado existen muchos sistemas gestores de bases de datos, a continuación se usará MySQL para realizar las prácticas, pero ello no significa ni que sea el mejor, o el más completo del mercado, sino más bien el más usado para aplicaciones web, de pequeña, mediana y, en algunos casos, de gran complejidad. Es un buen punto de partida y todos los conceptos en MySQL vistos son reutilizables en PosgresSQL, Oracle, u otro motor de bases de datos relacional.

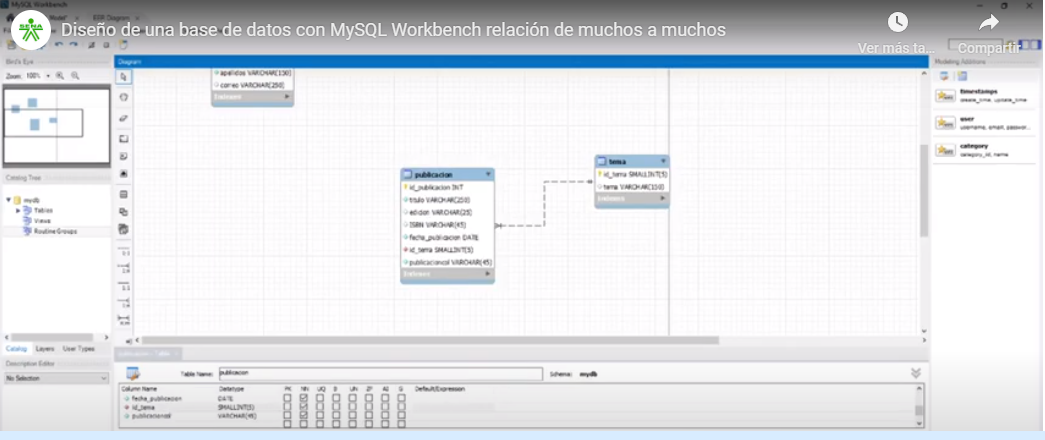
A continuación, se presenta el paso a paso de la descarga e instalación del entorno desarrollo WEB XAMPP:



Continuemos con la descarga en instalación de herramienta CASE MyQL *Workbench:*



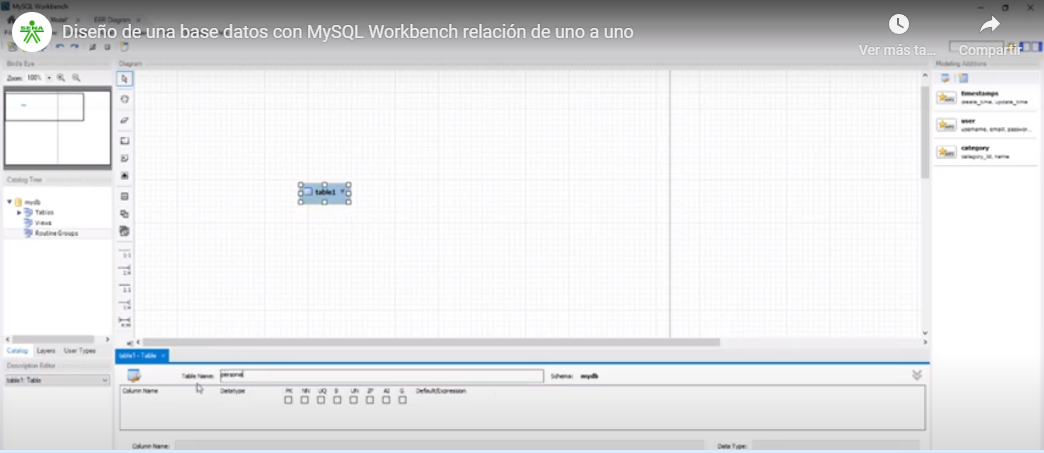
Ahora, estudiemos el diseño de una base de datos con MyQL *Workbench* relación de muchos a muchos:



Continuemos con el diseño de una base de datos con MyQL *Workbench* relación de uno a muchos:

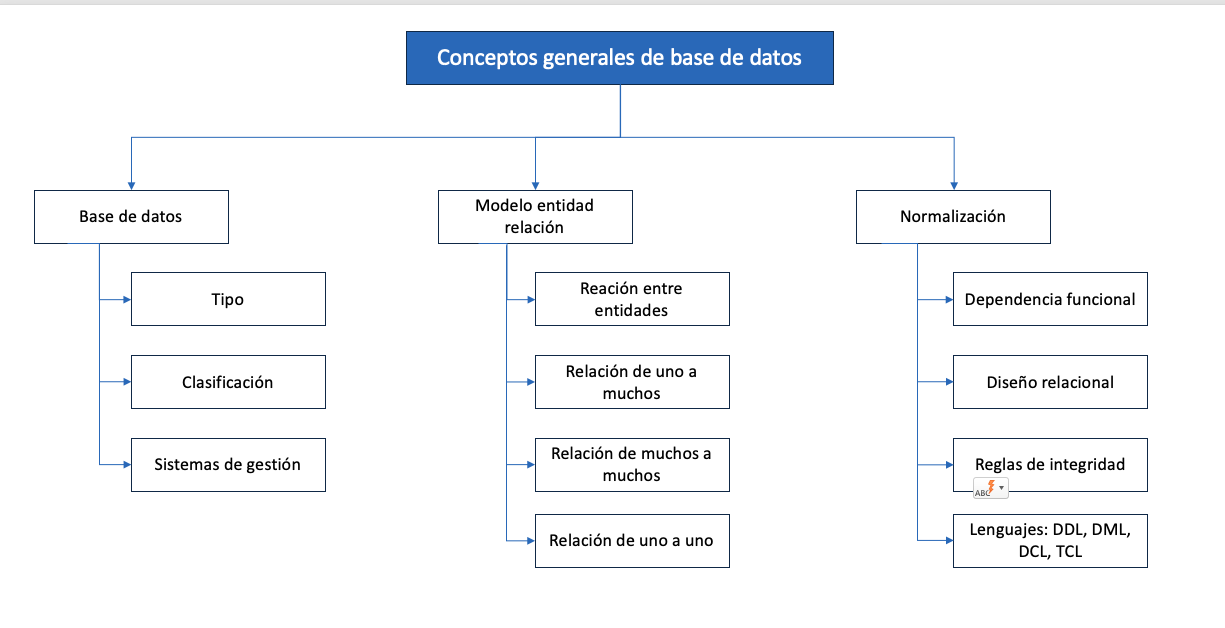


Y finalicemos con el diseño de una base de datos con MyQL *Workbench* relación de uno a uno:



1. **SÍNTESIS**

A continuación, se presenta una síntesis de la temática estudiada en el componente formativo:



1. **ACTIVIDADES DIDÁCTICAS (Se debe incorporar mínimo 1, máximo 2)**

|  |  |
| --- | --- |
| DESCRIPCIÓN DE ACTIVIDAD DIDÁCTICA | |
| Nombre de la Actividad | Tipos de bases de datos |
| Objetivo de la actividad | Identificar los diferentes tipos de bases de datos. |
| Tipo de actividad sugerida | Arrastrar y soltar |
| Archivo de la actividad  (Anexo donde se describe la actividad propuesta) | CF8\_Actividad\_didactiva.docx |

1. **MATERIAL COMPLEMENTARIO:**

|  |  |  |  |
| --- | --- | --- | --- |
| Tema | Referencia APA del Material | Tipo de material  (Video, capítulo de libro, artículo, otro) | Enlace del Recurso o  Archivo del documento o material |
| Nociones y modelos de bases de datos | Espinosa, A. P. (16 de noviembre de 2014). L6 1 Nociones y modelos de bases de datos. [Video] YouTube. | Video | https://www.youtube.com/watch?v=pPATLxijDfw |
| Fundamentos de bases de datos | Silberschatz, A., Korth, H. F., Sudarshan, S., Pérez, F. S., Santiago, A. I., y Sánchez, A. V. (2006). Fundamentos de bases de datos. McGrawHill. | PDF | <http://ceneval.isi.uson.mx/CENEVAL/4Implantacion%20de%20infraestructura/Fundamentos%20de%20base%20de%20datos.pdf> |

1. **GLOSARIO:**

|  |  |
| --- | --- |
| TÉRMINO | SIGNIFICADO |
| Base de datos: | conjunto ordenado y finito de operaciones que permite hallar la solución de un problema. |
| Diccionario de datos: | conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora. |
| Metadatos: | conjunto unitario de instrucciones que permite a una computadora realizar funciones diversas, como el tratamiento de textos, el diseño de gráficos, la resolución de problemas matemáticos, el manejo de bancos de datos. |

1. **REFERENCIAS BIBLIOGRÁFICAS:**

Asale y RAE. (2021). Diccionario de la lengua española. <https://dle.rae.es/base#CiiosqO>

Kyocera. (2021). Conceptos sobre base de datos orientada a objetos. Kyocera. <https://rb.gy/5deqdh>

López, J. C. (2009). Algoritmos y programación (guía para docentes). Eduteka <http://eduteka.icesi.edu.co/articulos/GuiaAlgoritmos>

Microsoft. (2021). Descripción de normalización de base de datos. Microsoft. <https://docs.microsoft.com/es-es/office/troubleshoot/access/database-normalization-description>

Resnick, M. (2007). Sembrando las semillas para una sociedad más creativa. Laboratorio de medios de MIT, Massachussets. Eduteka. <http://www.eduteka.org/ScratchResnickCreatividad.php>

Silberschatz, A., Korth, H. F., Sudarshan, S., Pérez, F. S., Santiago, A. I., y Sánchez, A. V. (2006). Fundamentos de bases de datos. McGrawHill <http://ceneval.isi.uson.mx/CENEVAL/4Implantacion%20de%20infraestructura/Fundamentos%20de%20base%20de%20datos.pdf>

1. **CONTROL DEL DOCUMENTO**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Nombre | Cargo | Dependencia | Fecha |
| Autor (es) | Henry Eduardo Bastidas Paruma | Experto Temático | Regional Cauca - Centro de Teleinformática y Producción Industrial |  |
| Ana Catalina Córdoba Sus | Evaluadora instruccional | Regional Antioquia - Centro de Servicios de Salud | Abril 2024 |
| Olga Constanza Bermúdez Jaimes | Responsable Línea de Producción Antioquia | Regional Antioquia - Centro de Servicios de Salud | Abril 2024 |

1. **CONTROL DE CAMBIOS**

**(Diligenciar únicamente si realiza ajustes a la Unidad Temática)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Nombre | Cargo | Dependencia | Fecha | Razón del Cambio |
| Autor (es) |  |  |  |  |  |