Sistemas de almacenamiento y modelado de datos

Breve descripción:

Este componente está orientado a brindar los conocimientos sobre el manejo de bases de datos, sentencias DML y sentencias SQL para realizar las consultas de la información almacenada, tablas y atributos, el esquema del modelo relacional y el diagrama entidad relación.

Mayo 2023

Grupo de Ejecución de la Formación Virtual

Tabla de contenido

Introducción

Se da la bienvenida a este componente formativo, en el cual se abordan las temáticas relacionadas con los sistemas de almacenamiento de información, también llamados bases de datos. A continuación, en el siguiente video, se presenta de modo genérico cómo ha evolucionado el proceso de almacenamiento y hemos llegado al el modelado de datos:

Video 1. Sistemas de almacenamiento y modelado de datos

Enlace de reproducción del video

Síntesis del video: sistemas de almacenamiento y modelado de datos

Sistema de almacenamiento y modelado de datos. Todos hemos en algún momento escuchado la palabra información y datos. Como todo lo relacionado con la tecnología, estos términos se relacionan con muchas cosas. En nuestro caso, los sistemas de almacenamiento, como es lógico, debemos saber que estos sistemas de almacenamiento de información irán cambiando a lo largo del tiempo.

Inicialmente, la información sólo podía ser almacenada en la memoria RAM de un computador, la cual solo permite el almacenamiento de información de una manera temporal, es decir, una vez que el programa deja de realizar su ejecución, los datos se pierden y se tienen que registrar nuevamente. Uno de los grandes esfuerzos en la historia fue cambiar esta dinámica, implementando el uso de ficheros o archivos en los cuales se almacenaba la información de forma permanente.

Pero con el tiempo, los ficheros se volvieron obsoletos, ya que almacenar mayor cantidad de volumen de información y estos no estaban en la capacidad de poder realizar dichas operaciones. Allí es cuando nace un proyecto que permitió la unificación de la información llamado bases de datos, el cual tiene como objetivo principal almacenar la información de manera ordenada, eficiente y que su acceso y administración sean de una manera mucho más sencilla y efectiva.

Los primeros sistemas de bases de datos se caracterizaron por solucionar los problemas de ese momento, pero después se hizo necesario que estos fueran evolucionando, ya que las demandas de manejo de información fueron cambiando de acuerdo con los procesos que las empresas requerían. Hoy día existen muchos motores de bases de datos, los cuales brindan, de acuerdo con las necesidades

técnicas de las organizaciones, soluciones que permiten gestionar y administrar la información de una manera mucho más eficiente que tiempos antiguos.

Uno de los aspectos que mayor importancia ha tenido es la capacidad de manejo de gran cantidad de transacciones por segundo. Se entiende por transacciones a la cantidad de solicitudes que puede realizar un usuario o un conjunto de usuarios al tiempo, sin afectar el rendimiento de los aplicativos o entrega de la información por parte del sistema.

Es importante tener en cuenta que existen algunas empresas que lideran el mercado de las bases de datos actualmente, como son Micro Systems con el motor de bases de datos Oracle, Microsoft con el motor de bases de datos SQL Server, y nuevamente Sun Micro Systems con la llamada MySQL. Este último ha tomado mucha popularidad en las comunidades de desarrollo y se encuentra presente como sistema de almacenamiento en muchos proyectos de software actualmente, y será nuestro motor seleccionado para el desarrollo de nuestro curso.

1. Conceptos de bases de datos

Las bases de datos se han convertido en la alternativa esencial en las empresas para el manejo y procesamiento de información, ya que estas tienen una capacidad de almacenamiento casi ilimitada y realizan los procesos de actualización y búsqueda de información con una velocidad y precisión casi perfecta.

Video 2. Conceptos base de datos

Enlace de reproducción del video

Síntesis del video: conceptos base de datos
Los sistemas de almacenamiento son capaces no solo de manejar grandes volúmenes de información, sino que también permiten tener modelos de seguridad

que permiten que la información sea almacenada de manera segura y eficiente. Desde su aparición, no han dejado de evolucionar y proponer mejores alternativas a empresas y usuarios en todo el mundo, destacando su capacidad de crecimiento y adaptación a las necesidades de hoy en día.

Las bases de datos nacen de la necesidad de poder almacenar información de manera segura, rápida y eficiente. En los antiguos sistemas de almacenamiento de información, el problema surgía en cómo controlar y mantener la eficiencia de las respuestas a las consultas realizadas en un sistema de almacenamiento. Los ficheros, que eran la primera opción desarrollada para manejar la información, se vieron afectados por el factor de manejo de grandes cantidades de datos e información simultánea.

Como respuesta a esto, nacieron las bases de datos como alternativa y como propuesta para un nuevo modelo de almacenamiento de información que permitiese que la información fuera almacenada de manera organizada y eficiente. Dentro del mundo empresarial, las bases de datos son las que se encargan de resguardar toda la información que una compañía puede generar. Por lo tanto, es posible observar que las bases de datos no solo solucionan problemas de manejo de información, sino que también se convierten en una estrategia de competitividad para cualquiera que recurra a ellas.

Las bases de datos proponen un estilo de almacenamiento de la información que se encuentra distribuido en filas y columnas. Este estilo de organización permite que la información se distribuya de tal manera que su crecimiento se realice de forma vertical.

Es así como los datos siempre incrementarán: entre mayor cantidad de filas contenga la base de datos, mayor cantidad de datos e información se verán reflejados en la misma.

Como se puede ver en la siguiente figura, los datos se encuentran registrados por medio de filas, las cuales tienen su incremento de forma vertical; es decir, cada vez que se registra una nueva persona, se agrega una nueva fila, que está acompañada del conjunto de columnas llamadas campos.

Figura 1. Estructura Base de Datos

En este caso, los campos que se observan son 5:

- PKID.
- Nombre.
- Contacto.
- FKCodigo_Tbl_Sexo.
- Edad.

Estos campos corresponden a los datos que se le solicitan al cliente y que luego pasan a ser parte de las filas del almacenamiento de la información en la base de datos. Esta debe contar con algo denominado llave primaria, la cual evita que un registro que se realizó previamente se pueda repetir en la base de datos, evitando así duplicidad de la información; para el caso de ejemplo, la llave primaria es PKID, que corresponde al número de identificación de la persona registrada.

Es importante saber que las bases de datos están conformadas por tablas, las cuales requieren que cada una de sus columnas (llamadas campos) reciban un tipo de dato específico que se almacene en él.

Los datos más conocidos y utilizados en los sistemas de bases de datos son los siguientes:

- Varchar
- Int
- Date
- Double
- Float

Estos tipos de datos permiten que la información se pueda almacenar en el formato que corresponda, de acuerdo con las necesidades del cliente. Se debe tener en cuenta que almacenar un dato que no corresponda a su valor dentro de la base de datos podría acarrear problemas de manipulación de los datos en un futuro, por eso se hace necesario que se determinen los tipos de datos de acuerdo con la información que será almacenada en el mismo.

2. Manejo de sentencias DML

Las sentencias DML, también conocidas como lenguaje de definición de datos, son utilizadas para realizar la manipulación de la información que se encuentra dentro de una base de datos y está inmersa en el uso de unos comandos particulares que permiten realizar estos procesos.

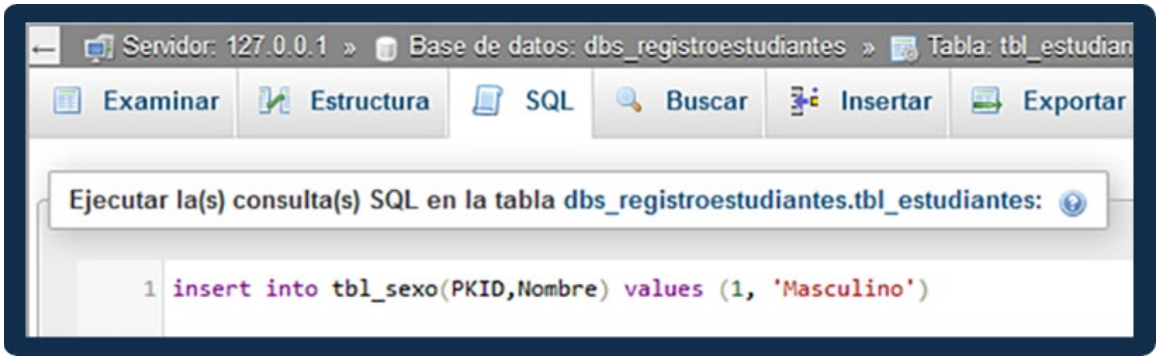
Para las empresas es muy importante tener un control y gestión de la información, lo cual implica realizar una serie de operaciones, las cuales comprenden agregar nuevos datos, eliminar datos registrados y actualizarlos.

Estas operaciones se realizan prácticamente todos los días en un sistema de almacenamiento de información, por eso, es importante el lenguaje DML, ya que este contiene las funciones específicas para realizar estos procesos, los cuales se examinan a continuación:

Comando insert

Esta instrucción se utiliza para generar un nuevo registro dentro de la base de datos y es importante tener en cuenta que los registros son líneas completas de forma horizontal; es decir, el crecimiento que esta va a tener permite que los datos se lean de forma vertical; cada fila se genera y luego debe ser recorrida de manera vertical para la recuperación de los datos, como se observa en la figura. Este es un ejemplo del uso del comando insert para crear un nuevo registro en la tabla de la base de datos:

Figura 2. Comando insert



Como se puede observar, en la imagen se aprecia el uso de la instrucción insert, la cual consta de la siguiente estructura: inicialmente, se llama al comando que permite crear el nuevo registro, llamado insert; luego, la palabra into, que indica dónde se va a colocar el nuevo registro; seguido del nombre de la tabla de la base de datos donde se insertarán los datos, acompañado de los campos de la tabla que se afectarán en el registro; luego, la palabra values, que significa valores a insertar, acompañada del orden específico de cómo se insertarán los datos en la tabla. Como se puede observar, primero se coloca el valor 1, que se guardará en el campo PKID, y luego, 'Masculino', que se guardará en el campo Nombre. Es importante tener en cuenta que cada dato a insertar está separado por coma, que los valores numéricos no llevan comillas y que los valores de carácter sí, como sucede en este caso.

La sentencia insert permite realizar un nuevo registro en la base de datos, de acuerdo con la estructura que se acaba de mencionar.

Cabe tener presente el orden de los campos a insertar, puesto que, si se invierten, estos no coincidirán y pueden mostrar errores al momento de la inserción de los datos. Se debe procurar que los datos que se inserten en la base de datos cumplan con las restricciones que se requieren.

Si se va a colocar un campo numérico, tener en cuenta que este se coloca literalmente como se escribe; mientras que si se va a insertar un dato de tipo carácter, se deben colocar las comillas, para evitar tener posteriores inconvenientes.

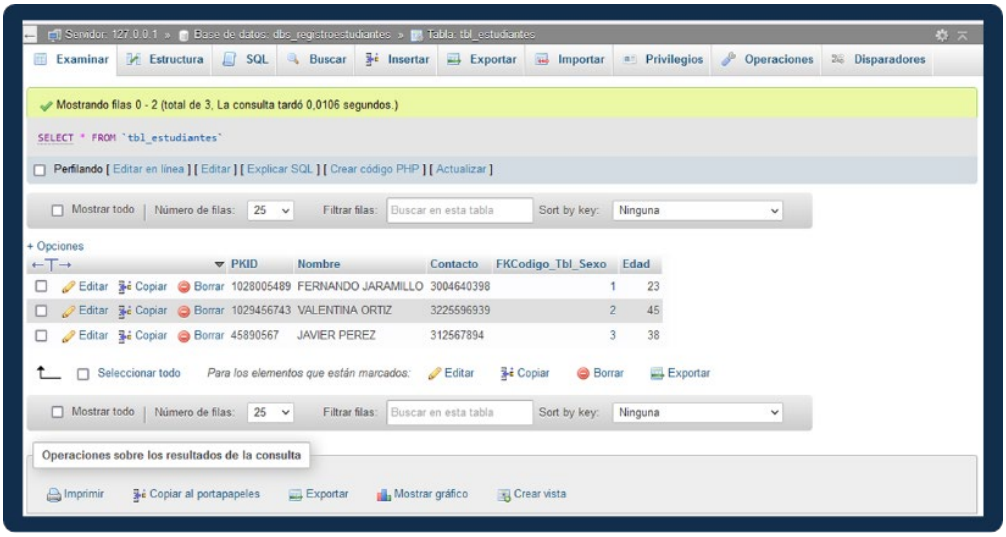
Comando update

Este comando se utiliza cuando se tiene un registro realizado y se desea hacer un cambio en una o varias de sus columnas.

Suponga que se registran los siguientes datos: Código, Nombre y Precio de un producto, y se requiere, en algún momento, cambiar el nombre del producto o su precio. El comando Update permite realizar esta operación para cambiar el valor contenido originalmente guardado en este campo.

Este comando es muy utilizado para realizar también cambios de varios registros; por ejemplo, que se cambie el precio de varios productos que comparten este mismo valor. A continuación, se visualiza la manera cómo el comando Update opera:

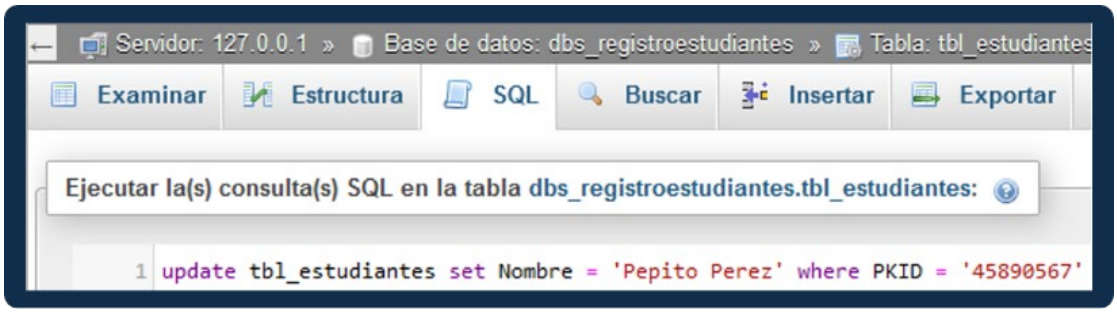
Figura 3. Actualización de registros



Como se puede observar, se tienen 3 registros, los cuales tienen los siguientes campos: PKID, Nombre, Contacto, FKCodigo_Tbl_Sexo, Edad.

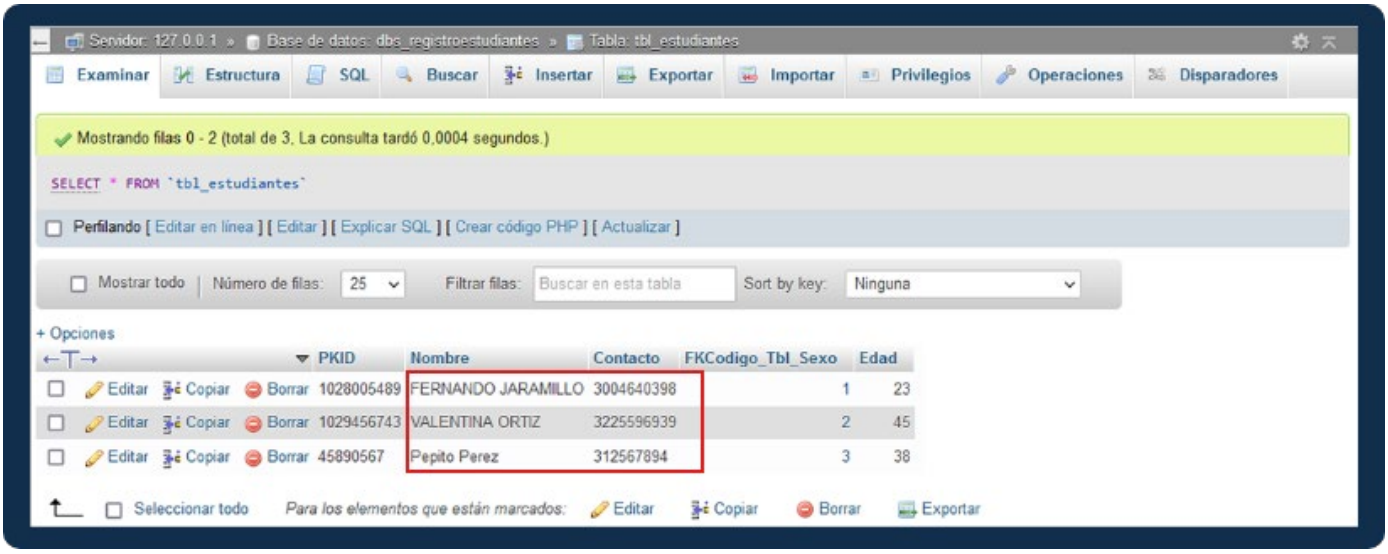
Cada uno de ellos guarda información del registro en columnas independientes, como se ilustra en la imagen; lo que se hará será actualizar uno de los datos que se encuentra en la tabla de la base de datos, en este caso, el que corresponde a “Javier Pérez”, y se ejecutará el comando, como se ilustra:

Figura 4. Comando update



Como se puede observar, hay que ubicarse en la pestaña SQL para ejecutar el comando de actualización del registro de la tabla, procediendo a actualizar el nombre de la persona registrada, con la condición del documento de identidad que se refleja en el campo PKID, ya que este es el valor de campo que no se repite y es único para cada registro, procediendo a presionar el botón Continuar, en la parte inferior, para ejecutar la estructura de comandos y obteniendo el siguiente resultado:

Figura 5. Resultados comando update



Como se puede observar en las imágenes anteriores, se ve el comportamiento de la actualización del registro de la base de datos con el nuevo nombre que se le ha asignado.

Es importante tener en cuenta la cláusula where, ya que, si no se especifica el registro particular con el código que se actualizará, este procederá a realizar la actualización de todos los registros de la tabla de la base de datos.

Comando delete

Este comando se utiliza para que, en el caso particular de que se necesite eliminar un registro de la base de datos, se pueda hacer con esta instrucción, aunque, en el caso particular de las bases de datos, la información que se registra dentro de ellas debe tener un tratamiento específico. En algunas ocasiones, realizar este proceso se hace bastante delicado, puesto que la eliminación de un registro implica su inexistencia y este podría estar relacionado con otros datos, lo cual podría generar un

caos en la base de datos; por eso, es importante verificar antes de realizar la eliminación.

Es posible que muchas de estas funcionalidades dentro de un sistema se manejen con mucha precaución debido al impacto negativo que puedan tener. A continuación, se muestra un ejemplo de lo que ocurriría.

Figura 6. Eliminar un dato de la Base de Datos

Como se puede evidenciar, se ha utilizado el comando "delete" para realizar la eliminación del registro de la base de datos.

Es importante denotar que para realizar la eliminación de este registro se ha colocado en la cláusula "where" el PKID o número de identificación, para señalar a la base de datos que este es el registro específico que debe eliminar.

Si este proceso no se realiza de esta manera, se estarían eliminando todos los registros de la base de datos, lo cual sería un problema. Es importante tener siempre presente que se debe especificar, mediante un campo llave primaria, el dato a eliminar, para que solo este registro sea el que se elimine de la base de datos.

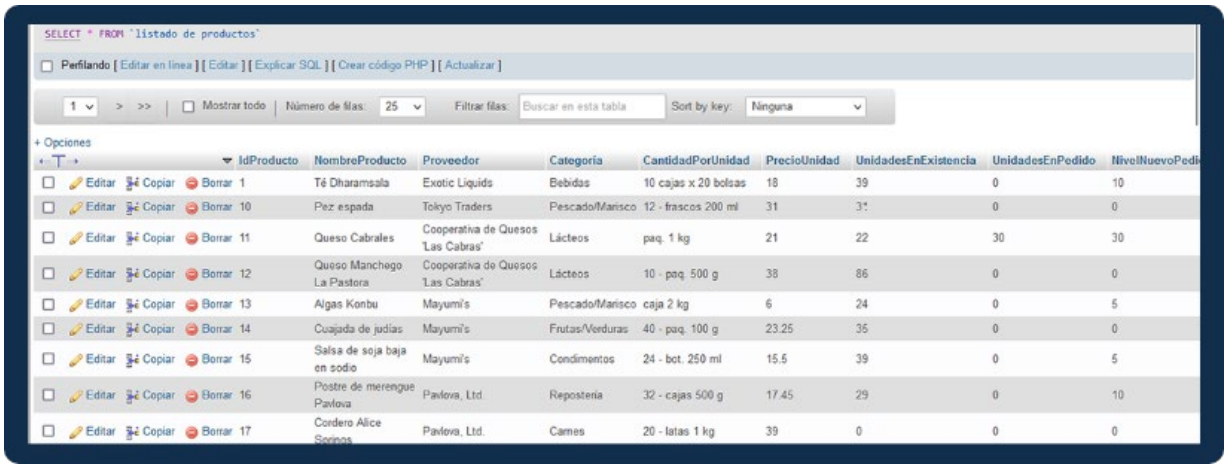
3. Manejo de sentencias SQL

Las sentencias SQL son instrucciones que permiten recuperar información que se encuentra previamente registrada en la base de datos, estas deben seguir un patrón y secuencia específica de sintaxis para poder realizar este proceso de manera correcta.

Una sentencia SQL también se puede comparar con una expresión ordenada que solicita información de manera específica de un sistema de almacenamiento, donde se indican las columnas o datos que se desean extraer y visualizar al usuario que requiere la información. Algo que es considerable dentro del uso de las sentencias SQL es la forma en cómo la información se solicita y los resultados que se esperan al realizar estas operaciones.

Se debe ser muy específico cuando estas se realizan. Dentro de las sentencias SQL, se encuentran varios comandos que acompañan las instrucciones y que permiten que la información pueda ser mucho más precisa y entendible para el usuario. A continuación, se observa una tabla de bases de datos con información y se ejecutan varias sentencias SQL para recuperar información de manera ordenada y precisa:

Figura 7. Tabla de contenidos



Como se puede observar, se tiene una tabla y, de acuerdo con esta información, se van a ejecutar las instrucciones SQL para recuperar los datos, teniendo en cuenta la secuencia que estos deben seguir para realizar dichos procesos; observe cómo:

Mostrar el nombre del producto y categoría de todos los que se encuentran registrados.

"select NombreProducto, Categoría from tbl_productos"

Se recibe la información acerca de un producto que se necesita. En la misma, se encuentra la cantidad, tiempo y aplicación.

En la consulta anterior, se percibe la aplicación de la sentencia SQL que permite recuperar el nombre y categoría de los productos registrados en la tabla llamada "tbl productos". En este caso, si el usuario requiere realizar dicho registro, este comando le permite solo recuperar estos datos de los registros de la base de datos.

Mostrar el nombre, existencias, id, precio unidad de los productos que tienen como unidades en existencias entre 10 y 50 productos.

"Select NombreProducto, IdProducto, PrecioUnidad from tbl_productos where UnidadesEnExistencia >= 10 and UnidadesEnExistencia <= 50"

Hay que recordar que estas instrucciones deben ser precisas al momento de solicitar la información, ya que si no se indica cuáles son los resultados, que se desea mostrarle al usuario, no se verán de la manera que se necesita.

Contar cuáles son los productos que se encuentran en la categoría de Bebidas y que su existencia esté dentro de 20 a 80, mostrar su id, nombre de producto, existencias, precio, proveedor y existencias.

"Select IdProducto,NombreProducto,UnidadesEnExistencia, PrecioUnidad,Proveedor, UnidadesEnExistencia from tbl_productos where Categoría = 'Bebidas' and (UnidadesEnExistencia >=20 and UnidadesEnExistencia <=80)"

En el resultado se especifica cada uno de los datos solicitados por el usuario, los cuales cumplen con el criterio de búsqueda dentro de la sentencia SQL. Los resultados presentados reflejan la solicitud explícita del usuario por la visualización de los mismos. Ahora bien, todo lo que se encuentra delante de la instrucción select y detrás de la instrucción from son los datos y encabezados que se visualizarán para el usuario en la instrucción SQL, por lo tanto, todos los campos que se soliciten en este apartado son los que serán visibles para el usuario una vez se ejecute la consulta SQL.

Seguramente, usted se hará la pregunta: ¿estos comandos en qué aportan a los sistemas de información? Pues hay que entender que el sistema de información se encuentra compuesto por varios componentes, uno de ellos llamado Front-End, que es toda la parte visual con la que interactúa el usuario; luego, existe un componente llamado Back-End, el cual define cuál es la lógica que se debe implementar en el sistema para que este cumpla las operaciones que requiere el usuario que el sistema realice.

"Ahora bien, en este punto, se puede cuestionar: ¿estos comandos en qué aportan a los sistemas de información? Pues hay que entender que el sistema de información se encuentra compuesto por varios componentes, uno de ellos llamado

Front-End, que es toda la parte visual con la que interactúa el usuario; luego, existe un componente llamado Back-End, el cual define cuál es la lógica que se debe implementar en el sistema para que este cumpla las operaciones que requiere el usuario que el sistema realice; es en esta última donde se codifican estos componentes.

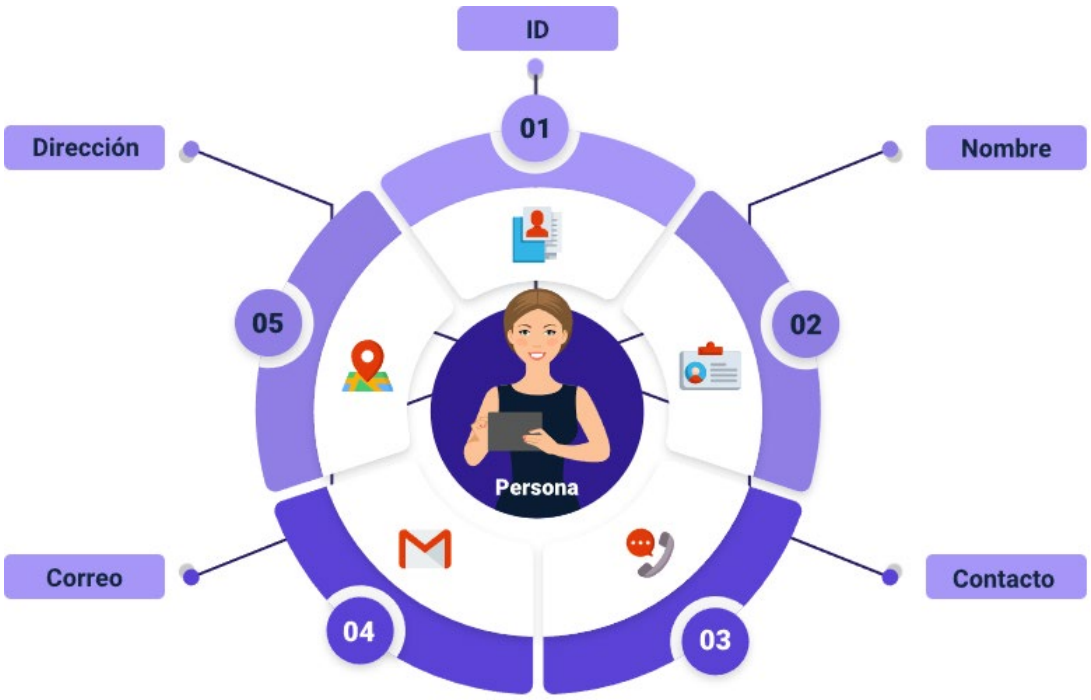
En pocas palabras, en el "back-end", existe una parte que permite la conexión con un sistema de bases de datos y recopila la información que solicita el usuario; este proceso es llevado a cabo a través de los comandos SQL. Dicho de otra forma, el usuario solicita la información interactuando con los componentes que el sistema tiene y luego este, al conectarse con la base de datos, solicita esta información a través de los comandos SQL, a la vez que estos devuelven al programa la información resultante solicitada por el usuario; siendo lo anterior, desde este punto de vista, relevante el estudio del lenguaje SQL, para poder comprender cómo la información llega desde el servidor de bases de datos a la pantalla del usuario mostrando la información que este requiere ver.

4. Modelado de datos

Un modelo de datos es un lenguaje orientado a hablar de una base de datos. Típicamente, permite describir las estructuras, el tipo de los datos y la forma en que se relacionan en una base.

El modelado de datos se utiliza para dar a entender los componentes y tipos de datos que se consagran dentro de la base de datos y que estos pueden, de alguna manera, ayudar no solo a la comprensión de la misma, sino también a entender cómo se pueden utilizar dichos datos al momento que se requieran modificar o utilizar dentro de la base de datos.

Figura 8. Modelado de datos



Como se puede observar en la imagen, se realiza el modelado de una entidad de la base de datos llamada Persona, la cual tiene los atributos: ID, Nombre, Contacto, Correo, Dirección.

Cada uno de estos campos permite establecer la información que se le solicita al cliente y cómo se almacenan los datos en la base. Cada campo de estos establece una característica que determina la información completa de la entidad, es decir, los agrega a todos dentro de un mismo espacio de almacenamiento.

El modelado de bases de datos es una técnica relevante en el ámbito del desarrollo de software, ya que permite conocer la distribución de los datos que se encuentran en la base de datos. También, establece la cantidad de datos a solicitar al usuario y, adicionalmente a ello, qué tipo de datos se solicitarán, los cuales pueden variar de acuerdo con las necesidades de almacenamiento que se tengan en el momento.

En una base de datos, es posible encontrar los siguientes tipos de datos:

- **Int.** Son datos que solo reciben números enteros y sin ningún tipo de separador, como puntos o comas.
- **Varchar.** Es un tipo de dato que permite recibir prácticamente cualquier dato, como números, texto, caracteres, símbolos, entre otros.
- **Float.** Es un tipo de dato que permite almacenar datos con puntos decimales, tales como: 1.5 o 4.8.

Estos son los tipos de datos principales que manejan las bases de datos, y, de acuerdo con el tipo de información a guardar, así será el tipo de dato a utilizar; por

ejemplo, si se requiere almacenar la información del documento de identidad, nombre, contacto y correo de una persona, quedarían distribuidos de la siguiente manera:

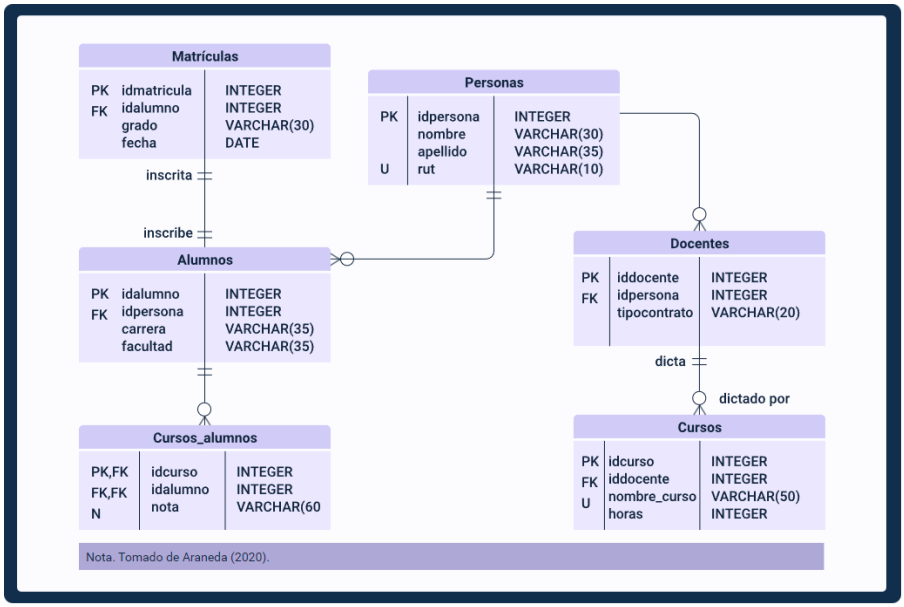
Tabla 1. Distribución según tipo de dato

Tipo de dato	Nombre del campo
Int	Identificación
Varchar	Nombre
Int	Contacto
Varchar	Correo

Es necesario tener en cuenta que, para el caso de los campos varchar, se debe estipular la longitud que estos deben tener, la cual está estrechamente relacionada con la cantidad de caracteres que ese campo recibirá. Por ejemplo: Nombre varchar(50), esta sería la manera correcta con la cual se establece la declaración de dicho campo y delimita la cantidad de caracteres que este puede tener internamente; para los campos int y float, no es necesario establecer una longitud, ya que por defecto el motor de la base de datos se los asigna.

A continuación, se observan otros tipos de diseños que integran la manera en cómo se puede visualizar la conexión de los datos dentro de la base de datos. Es importante tener en cuenta que se puede relacionar más de una entidad al tiempo para ilustrar la distribución de la información:

Figura 9. Relación de entidades



Nota. Tomado de Araneda (2020).

En este diseño, a diferencia del anterior, se visualizan los tipos de datos que tiene cada campo de la base de datos y su respectiva conexión.

En este caso, las relaciones están establecidas mediante llaves primarias y llaves foráneas; una llave primaria es aquella que convierte el valor que se almacena en ese campo como único y no repetible. Un ejemplo de ello es si se requiere almacenar la información de una persona, como: identificación, nombres y contacto; en este caso, la identificación se establecerá como llave primaria, para que el campo de identificación no se repita, ya que esto ocasionaría problemas de duplicidad de datos dentro de la entidad o tabla que almacene la información.

Por otro lado, las llaves foráneas permiten conectar las entidades; es decir, una llave primaria se conectará con una llave foránea para así relacionar las dos entidades y poder realizar el almacenamiento de la información. Cabe decir que todas las tablas o

entidades de una base de datos deben tener una llave primaria que identifique al campo que no se debe repetir.

Modelo físico de datos

El modelo de datos físicos representa cómo se construirá el modelo en la base de datos. Un modelo de base de datos físico muestra todas las estructuras de tabla, incluidos el nombre de columna, el tipo de datos de columna, las restricciones de columna, la clave principal, la clave externa y las relaciones entre las tablas.

El modelo físico de datos cuenta con características, etapas y funciones, las cuales se observan a continuación:

Características

- Cuenta con la especificación de todas las tablas y columnas.
- Las claves externas se usan para identificar relaciones entre tablas.
- La desnormalización puede ocurrir según los requisitos del usuario.

Etapas

- Convertir entidades en tablas.
- Convertir relaciones en claves externas.
- Convertir atributos en columnas.
- Modificar el modelo de datos físicos en función de las restricciones / requisitos físicos.

Funciones

- Permitir a los usuarios almacenar datos, acceder a ellos y actualizarlos, ocultando su estructura física.
- Proporcionar un catálogo (diccionario de datos) accesible por los usuarios.
- Proporcionar un mecanismo que garantice el procesamiento de las transacciones.
- Proporcionar un mecanismo que realice el control de la concurrencia.
- Proporcionar un mecanismo para recuperación ante fallos.
- Proporcionar un mecanismo de seguridad.
- Integrarse con algún software de comunicación.
- Encargarse de mantener las reglas de integridad.
- Encargarse de mantener la independencia entre los programas y la estructura de la base de datos.
- Proporcionar herramientas para administrar la base de datos

A su vez, a la hora del modelado de datos, debe procurar que cuente con lo siguiente:

- **Mínima redundancia de datos**. Al integrar los datos en una sola estructura lógica y almacenando cada ocurrencia de un ítem de dato en un solo lugar de la base de datos, se reduce la redundancia.
- **Consistencia de datos**. Al controlar la redundancia de datos, se reduce enormemente la inconsistencia, dado que, al almacenarse un dato en un solo lugar, las actualizaciones no producen inconsistencia.

- **Integración de datos**. En una base de datos, los datos son organizados de una manera lógica que permite definir los relacionamientos entre ellos.
- **Compartir datos**. Una base de datos es creada para ser compartida por todos los usuarios que requieran de sus datos; muchos sistemas de bases de datos permiten a múltiples usuarios compartirlas en forma concurrente, aunque bajo ciertas restricciones.
- **Esfuerzo por estandarización.** Establecer la función de Administración de Datos es una parte importante de este enfoque, su objetivo es tener la autoridad para definir y fijar los estándares de los datos, así como también posteriores cambios de estándares.
- **Facilitar el desarrollo de aplicaciones.** Este enfoque reduce el costo y tiempo para desarrollar nuevas aplicaciones.
- **Controles de seguridad, privacidad e integridad.** El control centralizado que se ejerce bajo este enfoque, a través de la función Administración de Base de Datos, puede mejorar la protección de datos en comparación con archivos tradicionales.
- **Flexibilidad en el acceso.** Este enfoque provee múltiples formas de recuperación de cada ítem de dato, permitiendo a un usuario mayor flexibilidad para ubicar datos que en archivos tradicionales.
- **Independencia de los datos.** Permite cambiar la organización de los datos sin necesidad de alterar los programas. Es uno de los objetivos principales del enfoque de base de datos.

- **Reducción de la mantención de programas.** Como los datos son independientes de los programas, se reduce la necesidad de modificar (mantener) los programas, aun cuando exista una modificación constante de estos.

5. Tablas y atributos

Las tablas, también llamadas entidades, son el espacio donde se almacena la información que registran los usuarios en un sistema de información.

Las tablas se dividen en los diferentes campos que convierten la información que suministra el usuario a través de los diversos canales de información. Se debe tener en cuenta que las tablas se conforman por atributos que, a su vez, permiten determinar:

- El tipo de dato
- La longitud
- Nombre de este (para identificarlo al momento de realizar una operación)

Ahora bien, en el caso de las tablas, estas pueden tener de manera inmersa una cantidad considerable de atributos, que permitirán que la información se almacene de manera separada dentro de la misma tabla; es decir, los datos se dividen por atributos, pero son almacenados en la tabla de la base de datos así:

Figura 10. Estructura de una tabla o entidad

En la anterior imagen, se puede observar la estructura de una tabla o entidad de bases de datos, la cual, en este caso, tiene varios atributos con tipos de datos específicos, los cuales almacenarán la información suministrada por el usuario.

Es importante tener en cuenta que los campos que son de tipo de datos "varchar" deben estar acompañados de la longitud máxima que este permite, pero los campos que son de tipo "int" no requieren de longitud al momento de registrarlos, ya que el sistema de bases de datos le asignará por defecto ese valor de manera automática.

Para llegar a comprender mejor las tablas, es necesario desarrollar varios conceptos que se deben manejar. Comenzando a partir del deseo de almacenar una información determinada en unos datos.

Los datos serían la información que deseamos almacenar. Un dato puede ser:

- El área de un parque natural
- El nombre de un parque natural
- La dirección de una oficina de correos
- El número de empleados de la oficina de correos
- El nombre de un accidente geográfico
- Las coordenadas de un accidente geográfico

Por otro lado, se tiene la entidad. Por entidad se entiende un objeto del mundo real que es posible distinguir del resto de objetos y del que interesan algunas propiedades.

En el modelo relacional, se puede observar que estas entidades se formarán por atributos o campos referidos a un mismo tema que interesa almacenar.

Una entidad debe definir cualquier objeto real o abstracto (que pueda ser pensado) y acerca del cual se quiere guardar información. Se representan mediante rectángulos en el modelo relacional.

Una entidad se correspondería, en el modelo relacional, con una tabla. La tabla, a su vez, estará formada por filas y columnas:

- **Filas:** cada unidad necesaria de almacenamiento, que se corresponden con los registros de la tabla.
- **Columnas:** hace referencia a los campos, unidad mínima de información, donde se podría almacenar cada dato referente a una propiedad del registro.

También sería posible tener una entidad débil, que es una entidad cuyos atributos no la identifican completamente, sino que solo la identifican de forma parcial. Esta entidad debe participar en una interrelación que ayuda a identificarla.

Es decir, la entidad débil tiene una dependencia funcional con otra entidad de la base de datos y está estrechamente relacionada con la conexión de las llaves que esta pueda tener; en otras palabras, depende de la existencia de otra entidad, la cual comparte con la misma la conexión de información registrada regularmente. Esta situación se da entre llaves primarias y foráneas donde el valor de la primaria debe coincidir completamente con el valor almacenado en la llave foránea.

A continuación se expone la representación de las entidades a través de un ejemplo:

Representación de las entidades

- **Persona.** La representación de la entidad débil es a través de un recuadro con doble línea.
- **Sexo_Persona.** La entidad “Persona” tiene dependencia con la entidad “Sexo” a través del atributo Sexo_Persona, ya que la entidad “Sexo” le proporciona este dato a la entidad “Persona”, dependiendo de esta asignación para poder realizar este registro.
- **Contacto.**
- **Correo.**
- **Dirección.**
- **ID.**
- **Nombre.**

Dos entidades

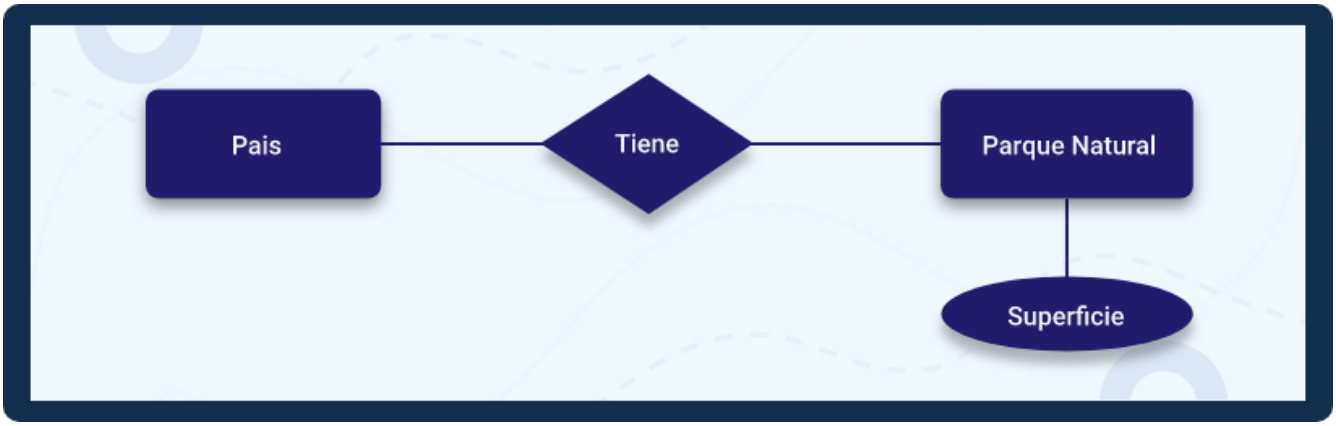
Se tienen dos entidades: una llamada “Persona” y otra llamada “Sexo”. Una débil y una fuerte.

- **Código.**
- **Descripción.**

Otro de los conceptos para tener en cuenta es el de atributo, los atributos son cada una de las propiedades o características que tiene un tipo de entidad o un tipo de relación. Toman valores de uno o varios dominios que representan las características de la entidad, y estos, en el momento de realizar el registro de la información, se convierten en los datos que se solicitan, aunque es importante tener en cuenta que los

atributos tienen diferentes tipos de datos, por lo cual, se hace necesario suministrar la información de acuerdo con este tipo de situaciones.

Figura 11. Atributos



Dentro del modelo relacional, se pueden encontrar atributos multivaluados y también opcionales, observe:

- **Atributo multivaluado**. Atributos de una entidad que pueden tener más de un valor.
- **Atributo optativo**. Aquel que puede admitir valores nulos.
- **Atributo identificador**. Uno o más campos cuyos valores son únicos en cada ejemplar de una entidad y deben distinguir a cada ejemplar que utiliza el modelo, teniendo en cuenta que todos los ejemplares de una entidad deben tener el mismo identificador.

Un atributo es importante aun cuando no tenga entidad concreta asociada, entonces se trata de una entidad y no de un atributo.

Los atributos y tablas son componentes que le dan vida y razón de ser a la base de datos, estos permiten organizar la estructura de la base de datos de manera correcta.

Los atributos de una entidad reflejan la cantidad de datos que esta recibe, la longitud y tipo de dato que es compatible con el mismo; es decir, si un atributo tiene como tipo de dato entero o "`int`", no se podrán almacenar caracteres ni ningún tipo de dato diferente en el mismo.

Ahora bien, cabe destacar que, si el tipo de dato utilizado en el atributo es de tipo texto, carácter o "`varchar`", este permitirá almacenar cualquier tipo de dato que se requiera, solo que es necesario ilustrar la cantidad de datos que tiene para poder indicarle al usuario cómo debe ingresar los datos de manera correcta.

6. Modelo relacional

Para mejor ilustración de lo que es un modelo relacional y complementar lo que ya se ha visto hasta ahora sobre el mismo, es preciso comenzar con un ejemplo simple de dos tablas que una pequeña empresa podría utilizar para procesar pedidos de sus productos.

La primera tabla es una tabla de información del cliente, por lo que cada registro incluye el nombre, la dirección, la información de envío y facturación, el número de teléfono y otra información de contacto. Cada bit de información (cada atributo) está en su propia columna y la base de datos asigna un ID único (una clave) a cada fila.

En la segunda tabla, una tabla del pedido del cliente, cada registro incluye el ID del cliente que realizó el pedido, el producto solicitado, la cantidad, el tamaño y el color seleccionados, etcétera, pero no el nombre o la información de contacto del cliente.

Estas dos tablas tienen solo una cosa en común: la columna de ID (la clave).

Gracias a esa columna en común, la base de datos relacional puede crear una relación entre las dos tablas. Entonces, cuando la aplicación de procesamiento de pedidos de la empresa envía un pedido a la base de datos, la base de datos puede ir a la tabla de pedidos del cliente, extraer la información correcta sobre el pedido del producto y usar el ID del cliente de esa tabla para buscar la información de facturación y envío en la tabla de información del cliente.

Entonces, el almacén puede extraer el producto correcto, el cliente puede recibir la entrega del pedido a tiempo y la empresa puede recibir el pago.

El modelo relacional significa que las estructuras lógicas de datos (las tablas de datos, las vistas y los índices) están separadas de las estructuras físicas de almacenamiento.

Esta separación significa que los administradores de bases de datos pueden administrar el almacenamiento físico de datos sin afectar el acceso a esos datos como una estructura lógica. Por ejemplo, cambiar el nombre de un archivo de base de datos no cambia el nombre de las tablas almacenadas en él.

La distinción entre lógica y física también se aplica a las operaciones de la base de datos, que son acciones claramente definidas que permiten a las aplicaciones manipular los datos y las estructuras de la base de datos. Las operaciones lógicas permiten que una aplicación especifique el contenido que necesita, mientras que las operaciones físicas determinan cómo se debe acceder a esos datos y luego realizan la tarea.

Para garantizar que los datos sean siempre precisos y accesibles, las bases de datos relacionales siguen ciertas reglas de integridad. Por ejemplo, una regla de integridad puede especificar que no se permitan filas duplicadas en una tabla, para eliminar la posibilidad de que se ingrese información errónea en la base de datos, como se ilustra en la imagen a continuación.

Figura 12. Modelo relacional

El siguiente video expone cómo se llega al modelo relacional:

Video 3. ¿Cómo nace el modelo relacional?

Enlace de reproducción del video

Síntesis del video: ¿Cómo nace el modelo relacional?
En los primeros años de las bases de datos, cada aplicación almacenaba datos en su propia estructura única. Cuando los desarrolladores querían crear aplicaciones para usar esos datos, tenían que saber mucho sobre la estructura de datos particular para encontrar los datos que necesitaban. Estas estructuras de datos eran ineficientes, difíciles de mantener y difíciles de optimizar para ofrecer un buen rendimiento de la aplicación. El modelo de base de datos relacional se diseñó para resolver el problema de varias estructuras de datos arbitrarias. El modelo de datos relacionales proporcionó una forma estándar de representar y consultar datos que cualquier aplicación podría utilizar desde el principio. Los desarrolladores reconocieron que la principal fortaleza del modelo de base de datos relacional estaba en el uso de tablas, que era una forma intuitiva, eficiente y flexible de almacenar y acceder a información estructurada. Con el tiempo, cuando los desarrolladores comenzaron a utilizar el lenguaje de consulta SQL para escribir y consultar datos en una base de datos, surgió otra fortaleza del modelo relacional. Durante muchos años, se utilizó SQL como lenguaje para consultas de bases de datos. Este se basa en el álgebra relacional y proporciona un lenguaje matemático internamente consistente que facilita la mejora del rendimiento de todas las consultas de la base de datos. Organizaciones de todo tipo y tamaño utilizan el modelo relacional, simple pero poderoso, para una amplia variedad de necesidades de información. Las bases

de datos relacionales se utilizan para hacer seguimiento de los inventarios, procesar transacciones de comercio electrónico, administrar grandes cantidades de información de clientes de misión crítica y mucho más. Se puede considerar una base de datos relacional para cualquier necesidad de información en la que los puntos de datos se relacionan entre sí y se deban administrar de una manera segura, consistente y basada en reglas.

Las bases de datos relacionales existen desde la década de 1970. Actualmente, las ventajas del modelo relacional continúan convirtiéndolo en el modelo más aceptado para bases de datos.

Modelo relacional y coherencia de datos

El modelo relacional es el mejor para mantener la consistencia de los datos en todas las aplicaciones y copias de la base de datos (denominadas instancias).

Por ejemplo, cuando un cliente deposita dinero en un cajero automático y luego mira el saldo de la cuenta en un teléfono móvil, el cliente espera ver que ese depósito se refleje inmediatamente en un saldo de cuenta actualizado. Las bases de datos relacionales se destacan en este tipo de consistencia de datos, lo que garantiza que múltiples instancias de una base de datos tengan los mismos datos todo el tiempo.

Es difícil para otros tipos de bases de datos mantener este nivel de coherencia oportuna con grandes cantidades de datos. Algunas bases de datos recientes, como NoSQL, solo pueden proporcionar “coherencia eventual”.

Según este principio, cuando se escala la base de datos o cuando varios usuarios acceden a los mismos datos al mismo tiempo, los datos necesitan algo de tiempo para “ponerse al día”. La coherencia eventual es aceptable para algunos usos, como mantener listados en un catálogo de productos, pero para operaciones comerciales críticas, como transacciones de carrito de compras, la base de datos relacional sigue siendo lo ideal.

7. Diagrama entidad-relación

El Modelo Entidad Relación (MER) es una herramienta de modelado que fue introducida originalmente por Peter Chen, en 1976, y aunque ha sufrido variaciones en cuanto a los elementos de diagramas utilizados para representar sus elementos, su operación y utilidad siguen vigentes.

La base del MER está en identificar los elementos o entidades importantes del sistema, los datos (atributos) que componen cada uno de ellos y la interacción (relación) entre dichos elementos.

El MER consiste en representar, a nivel conceptual, los datos que soportan el funcionamiento de un sistema y sus componentes básicos son: Entidades, Atributos y Relaciones.

Las entidades representan abstracciones con atributos que almacenan datos; las relaciones son las asociaciones que existen entre entidades y permiten generar información al combinarse entre sí.

Hasta este momento ya se ha dado una introducción a estos componentes básicos, pero para profundizar se amplía sobre su conceptualización a continuación:

Entidad. Se denomina entidad a todo ente (conceptual o físico) del cual se desea establecer su participación dentro de un sistema de información. Una entidad concreta o física es aquella con existencia física, representa un objeto del mundo real (persona o elemento). Una entidad abstracta no tiene una representación física concreta (posición laboral, asignatura).

Atributo. El atributo es un elemento de información que caracteriza a una entidad, identificándola, calificándola, cuantificándola, o declarando su estado. Por lo general, una entidad se compone de uno o más atributos (edad, genero, estatura, nombre, etc.). Los atributos permiten diferenciar elementos dentro de un conjunto de entidades. Dentro de una entidad de tipo persona, es muy raro el caso que existan dos con exactamente los mismos atributos.

Relaciones. Las relaciones identifican la interacción que existe entre dos o más entidades. Establecen el comportamiento del sistema de información.

Los elementos básicos de MER se presentan en un diagrama simple, que permite establecer en forma general un modelo de datos, tal como se ejemplifica en la imagen:

Figura 13. Elementos básicos Diagrama Entidad - Relación

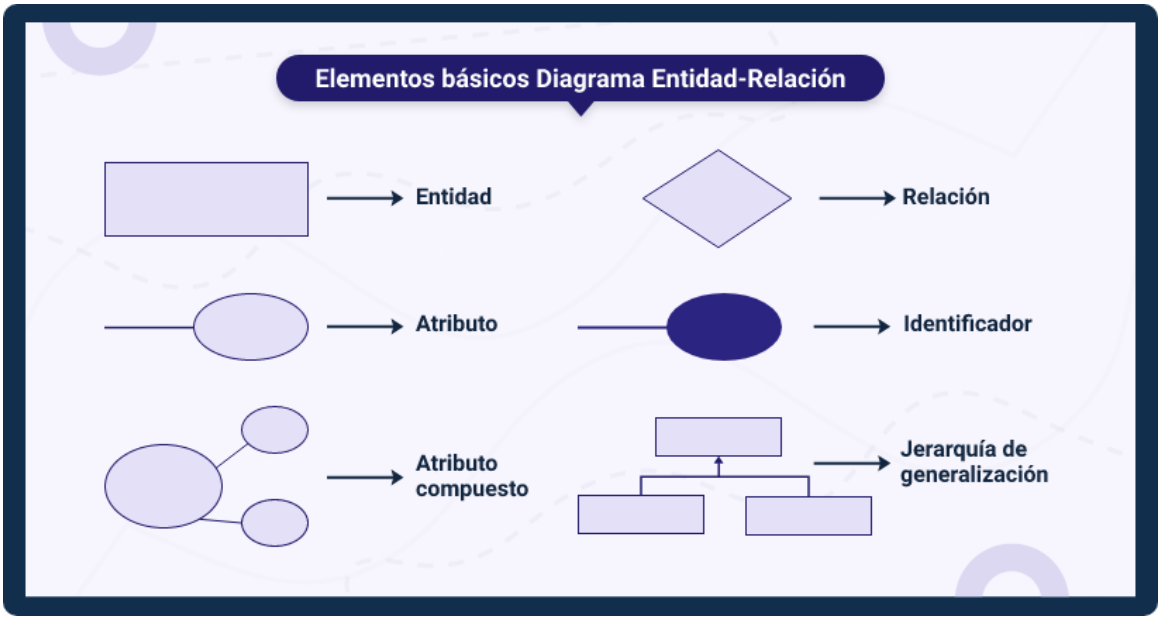
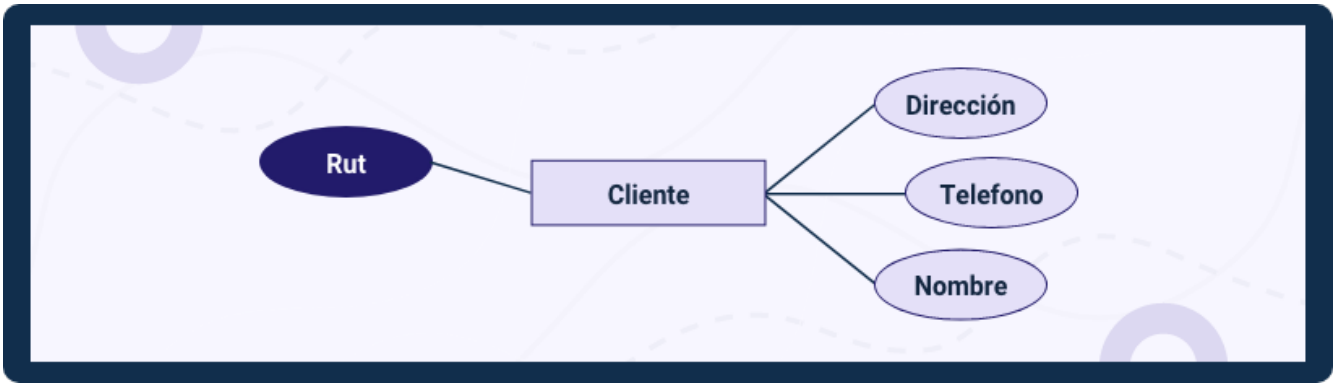


Figura 14. Un ejemplo de uso sería el siguiente:



Modelamiento lógico

Un modelo de datos lógicos describe los datos con el mayor detalle posible, independientemente de cómo se implementarán físicamente en la base de datos.

El mismo también cuenta con características propias y etapas para realizar, las cuales se muestran a continuación:

Tabla 2. Características y etapas del modelamiento lógico

Características	Etapas
• Incluye todas las entidades y relaciones entre ellos. • Todos los atributos para cada entidad están especificados. • La clave principal para cada entidad está especificada.	• Especificar claves primarias para todas las entidades. • Encontrar las relaciones entre diferentes entidades. • Encontrar todos los atributos para cada entidad. • Resolver las relaciones de muchos a muchos.

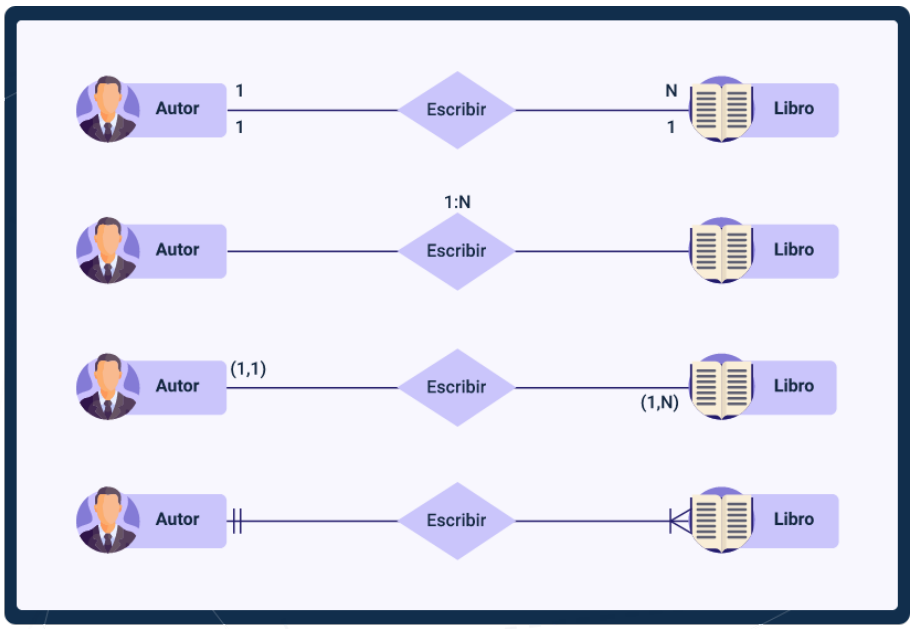
• Se especifican las claves externas (claves que identifican la relación entre diferentes entidades). • La normalización ocurre en este nivel.	• Realizar proceso de normalización.

Cardinalidad

La cardinalidad está definida como la cantidad de elementos, en términos de proporción, que participan en la relación entre dos o más entidades. Esta puede ser entre elementos únicos (unitarios) o múltiples. Generalmente, se utiliza la denominación “1” para elementos unitarios y “N” para varios elementos participantes.

Por ejemplo, el diagrama siguiente expresa que “un autor escribe varios libros” o “un libro es escrito por un autor”:

Figura 15. Ejemplo cardinalidad

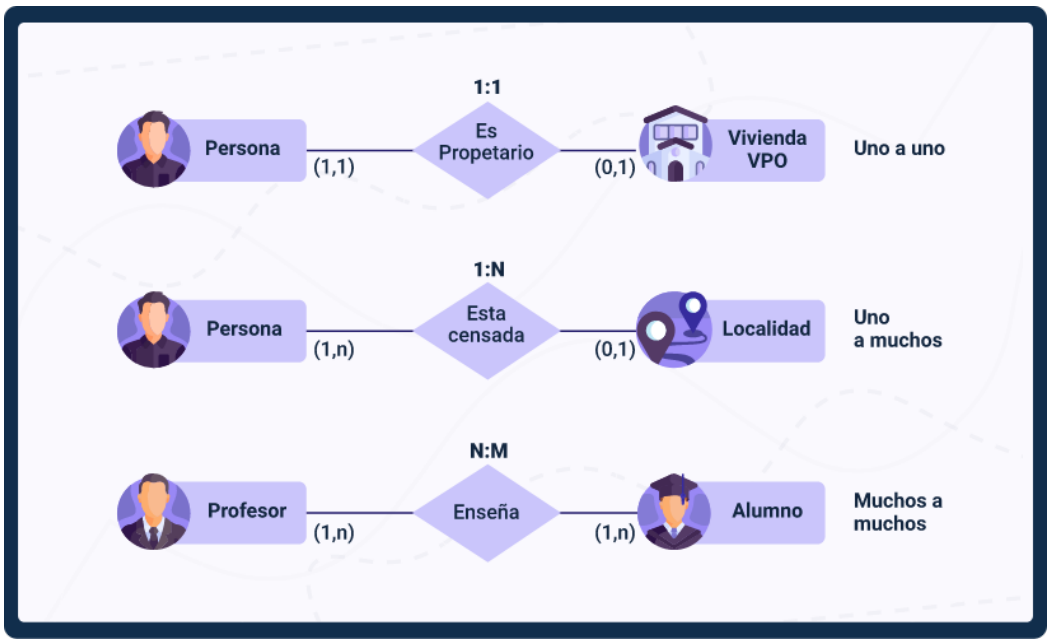


Las expresiones que pueden indicarse a partir de estos 4 diagramas son:

- Un Autor escribe Al Menos un Libro (Cardinalidad mínima=1)
- Un Autor escribe Varios Libros (Cardinalidad máxima = N)
- Un Autor escribe uno o Varios Libros (Cardinalidad mínima= 1, máxima =N)
- Un Libro es escrito por un solo un Autor (Cardinalidad mínima y máxima = 1)

Otro ejemplo que se podría traer como práctica es el siguiente, donde se desea colocar la cardinalidad correcta, de acuerdo con el tipo de relación que se refleja dentro de la base de datos, y es posible observarlo en la figura siguiente:

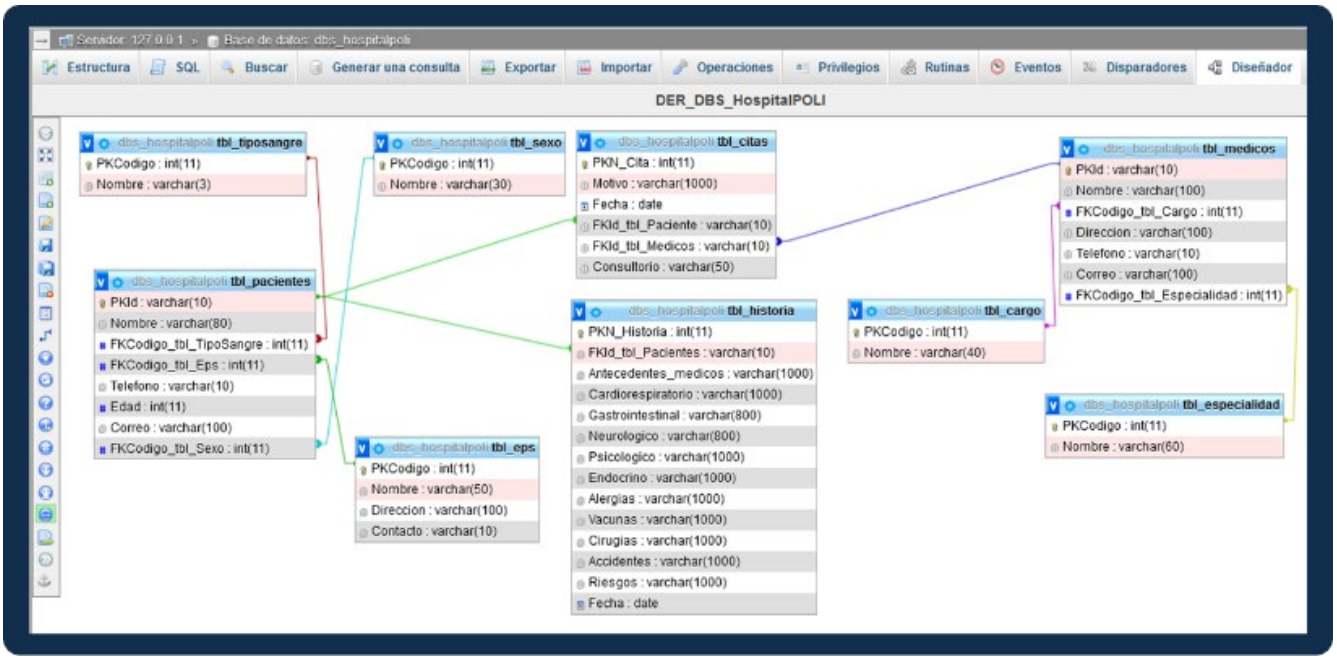
Figura 16. Ejemplo de cardinalidad según el tipo de relación



Una vez ya se han visto reflejados los datos dentro de un sistema de bases de datos real, en este caso, se está utilizando el motor de base de datos MySQL.

Es posible observar cómo quedan distribuidas las relaciones de acuerdo con las conexiones que se establecen en la base de datos; es decir, se pasa del modelo conceptual de entendimiento básico a ya crear un sistema de bases de datos con todas sus relaciones completas, como se puede visualizar a continuación:

Figura 17. Ejemplo de sistema de bases de datos



Como se puede observar en la imagen anterior, se visualiza la forma en cómo está distribuida la información con las diferentes entidades que hacen parte de esta.

También se pueden visualizar los diferentes tipos de datos que tienen los atributos de la base de datos. Es necesario recordar que para relacionar dos entidades estas utilizan algo llamado llaves, las cuales pueden ser primarias o foráneas, y deben compartir la misma longitud y tipo de dato para poder relacionarse de manera correcta.

En pocas palabras, si se van a relacionar 2 entidades y una tiene la llave de tipo “int” y la otra de tipo “varchar”, no se podrá realizar la conexión de las entidades, ya que

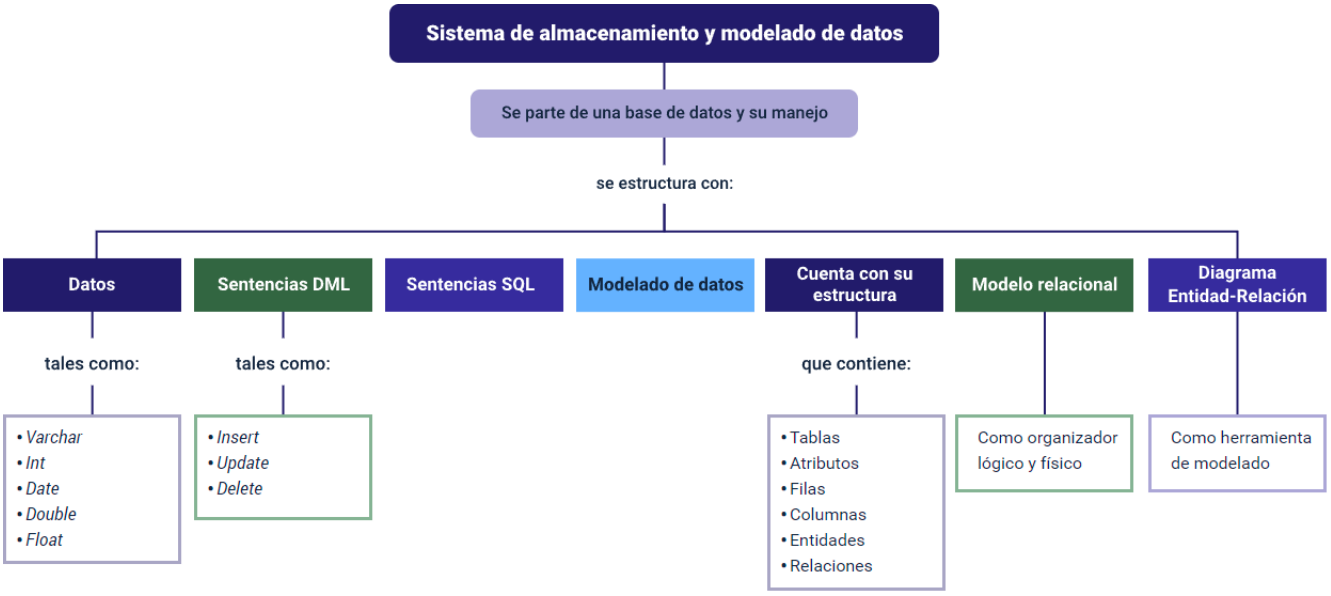
estas tienen como restricción el tipo de dato y longitud para poder realizar el proceso de conexión de manera correcta.

Síntesis

Las bases de datos son esa alternativa esencial en las empresas para el manejo y procesamiento de información, por su capacidad de almacenamiento casi ilimitada y posibilidades de realizar los procesos de actualización y búsqueda de información con una velocidad y precisión casi perfectas.

Por lo cual, al hablar de sistemas de almacenamiento y modelado de datos, se hace referencia expresamente a ellas, a sus elementos constitutivos y a las sentencias, modelos y diagramas que se desplegarán a través de estas.

A continuación, una síntesis de lo visto en este componente:



Material complementario

Tema	Referencia APA	Tipo de material	Enlace del Recurso
3. Manejo de sentencias SQL	Chávez, J. (2016). 7 video consultas en phpMyAdmin con sql [Video]. YouTube.	Video	https://www.youtube.com/watch?v=fHU0lY3FqB0&ab_channel=J%C3%B3seLuisCh%C3%A1vezG%C3%B3mez
6. Modelo relacional	Coronado, D., Rodríguez, R., Tineo, L. y Carrasquel, S. (2018). Gestión de Datos Difusos: Atributos Tipo 2 y Tipo 3 en bases de datos relacionales. Publicaciones en Ciencias y Tecnología, 12(2), p. 83-95.	Artículo	https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1i756fj/TN_cdi_doaj_primary_oai_doaj_org_article_b4db22235526430295652b34b49ad6dd
6. Modelo relacional	Aragón, Y., González, C., Hernández, O. y Hernández, E. (2018). Herramienta para el aprendizaje de bases de datos relacionales. Revista Cubana De Ciencias Informáticas, 12(3), p. 163-176.	Artículo	https://www.youtube.com/watch?v=cYLapo1FFmA

Glosario

Atributo: son características de las entidades que permiten asignar un nombre, longitud y tipo a la misma y reflejan la información que se suministra a la base de datos.

Base de datos: es una estructura de almacenamiento de información permanente y organizada, que permite insertar, actualizar, buscar y eliminar datos con facilidad.

Cardinalidad: son las restricciones que tienen las relaciones de las entidades en la base de datos.

DDL: significa lenguaje de definición de datos y se utiliza para crear la estructura principal de la base de datos, tales como las tablas y las relaciones.

DML: significa lenguaje de manipulación de datos y permite realizar la inserción, actualización y eliminación de datos en una base de datos.

Entidad: es un componente de la base de datos que permite el almacenamiento de la información que se le suministra a la base de datos.

Modelo entidad-relación: es un esquema que refleja la manera como están relacionados los datos dentro de la base de datos.

Referencias bibliográficas

Araneda, P. (2020). Base de Datos. El Camino de los datos a la información. Bookdown. https://bookdown.org/paranedagarcia/database/modelamiento-de-datos.html

Créditos

Nombre	Cargo	Regional y Centro de Formación
María Camila García Santamaria	Líder del equipo	Dirección General
Rafael Neftalí Lizcano Reyes	Asesor metodológico y pedagógico	Centro Industrial del Diseño y la Manufactura - Regional Santander
Dulfran Antonio Montaño Montaño	Experto temático	Centro De Diseño Y Metrología - Regional Distrito Capital
Zvi Daniel Grosman Landáez	Diseñadora instruccional	Centro de Gestión Industrial - Regional Distrito Capital
Andrés Felipe Velandia Espitia	Asesor metodológico	Centro de Diseño y Metrología - Regional Distrito Capital
Darío González	Corrector de estilo	Centro de Diseño y Metrología - Regional Distrito Capital
Yerson Fabian Zarate Saavedra	Diseñador web	Centro Industrial del Diseño y la Manufactura - Regional Santander
Camilo Bolaño	Desarrollador fullstack	Centro Industrial del Diseño y la Manufactura - Regional Santander
Zuleidy María Ruiz Torres	Validación y vinculación en plataforma LMS	Centro Industrial del Diseño y la Manufactura - Regional Santander
Wilson Andrés Arenales Cáceres	Ilustración	Centro Industrial del Diseño y la Manufactura - Regional Santander
Mary Jeans Palacio Camacho	Producción audiovisual	Centro Industrial del Diseño y la Manufactura - Regional Santander
Carlos Eduardo Garavito Parada	Producción audiovisual	Centro Industrial del Diseño y la Manufactura - Regional Santander

Luis Gabriel Urueta Alvarez	Validación y vinculación en plataforma LMS	Centro Industrial del Diseño y la Manufactura - Regional Santander
Daniel Ricardo Mutis Gómez	Validación de contenidos accesibles	Regional Santander - Centro Industrial del Diseño y la Manufactura