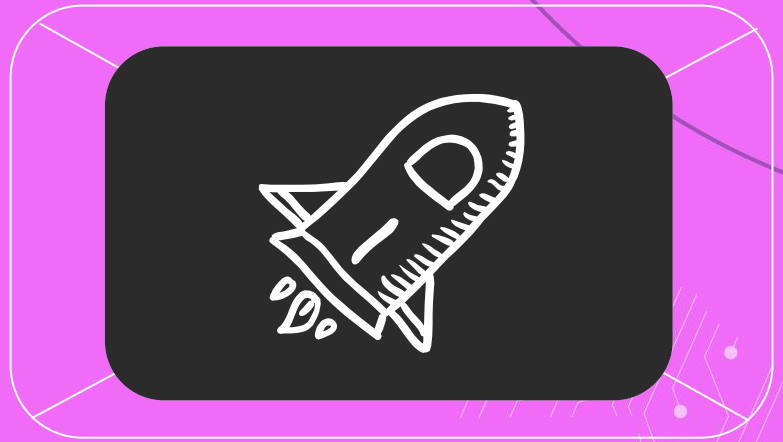
Gobierno del CHACO | Ministerio de la Producción y el Desarrollo Económico Sostenible

INFORMATORIO

Semana 6
¡Bienvenidos!



Temas de hoy

1. Bases de Datos: Introducción SQL, BD Relacionales y No Relacionales.
2. Modelos de Datos: Entidades, Atributos, Relaciones, Modelo MER.

1

Bases de Datos

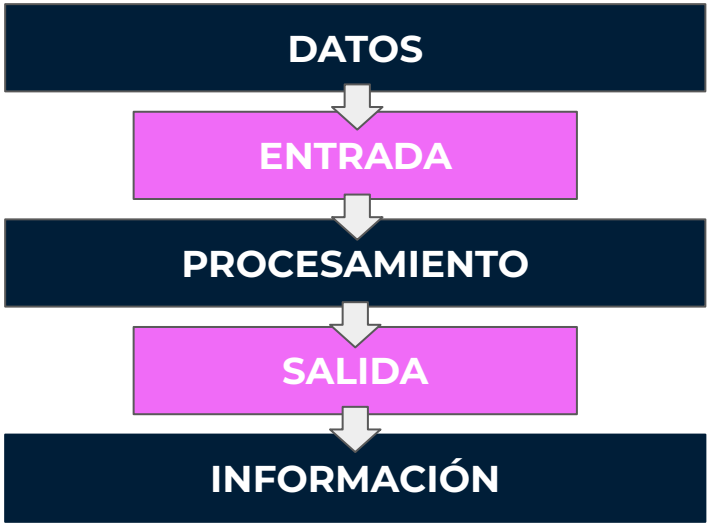
El uso de variables y estructuras de datos no nos permite mantener un registro de varios datos a largo plazo, ya que, los datos almacenados existen sólo mientras se esté ejecutando nuestro algoritmo.

Si queremos guardar datos de manera persistente, utilizamos bases de datos que nos permitirán organizar y almacenar información que sea trazable en el tiempo. 🗄️ 👉

Bases de datos: Introducción

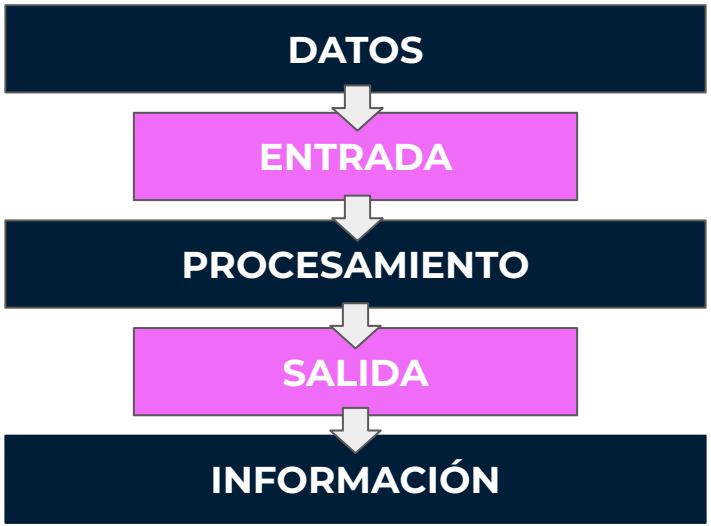
Introducción

Las bases de datos son un elemento fundamental en el entorno informático hoy en día y tienen aplicación en prácticamente la totalidad de campos.
En el entorno del mercado actual, la rapidez en la recopilación y análisis de los datos de una empresa son imprescindibles para su éxito. Para conseguirlo existe cada vez una mayor demanda de datos y, por tanto, más necesidad de gestionarlos. Esta demanda siempre ha estado presente en empresas y sociedades, pero en estos últimos años, se ha disparado debido al acceso multitudinario a las redes integradas en Internet y a la aparición de los dispositivos móviles que también requieren esa información.



Para poder entender mejor el concepto de Bases de Datos es importante que veamos como es el flujo de información en un sistema o proceso.

Bases de datos: Introducción



El procesamiento de datos consiste en transformar datos puros en información organizada, significativa y útil para luego almacenarlos.
En el proceso de flujo de información es importante identificar los conceptos de:

→ **DATO**: Se refiere a hechos, eventos, transacciones que han sido registrados. Es la entrada sin procesar de la cual se produce la información.

- **INFORMACIÓN**: Se refiere a los datos que han sido procesados y comunicados de tal manera que pueden ser entendidos e interpretados por el receptor.
Los datos no tienen sentido por sí mismos, pero al ser procesados y contextualizados se convierten en información certera y disponible para conocer un fenómeno, tomar decisiones o ejecutar acciones.

Bases de datos

Definición

Una Base de Datos (BD o database) es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. Los datos almacenados se encuentran estructurados y relacionados entre ellos.

Permite al usuario manipular la información:

- Insertar nuevos datos
- Recuperar y actualizar datos

Entre las principales características de los sistemas de base de datos podemos mencionar:

- Independencia lógica y física de los datos.
- Acceso a través de lenguajes de programación estándar.
- Redundancia mínima.
- Acceso concurrente por parte de múltiples usuarios.
- Integridad de los datos.
- Seguridad de acceso y auditoría.
- Respaldo y recuperación.

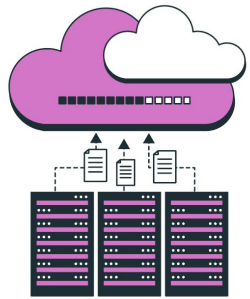
Importancia de las Bases de Datos

Las Bases de Datos (BD) son herramientas fundamentales para el desarrollo de software y la gestión de información en empresas y organizaciones. Las BD permiten almacenar y organizar datos de manera eficiente, centralizada y segura, y facilitan el acceso rápido a la información desde cualquier lugar.
Entre las ventajas del uso de BD podemos mencionar:

- Control sobre la redundancia de datos
- Consistencia de datos
- Mejora en la integridad de datos
- Mejora en la accesibilidad a los datos
- Aumento de la concurrencia

Entre las desventajas del uso de Bases de Datos tenemos:

- **Complejidad**
Es preciso comprender muy bien el uso de los gestores de Base de Datos para poder realizar un buen uso de ellos.



- **Costo del equipamiento adicional**
Pueden hacer que sea necesario adquirir más espacio de almacenamiento.

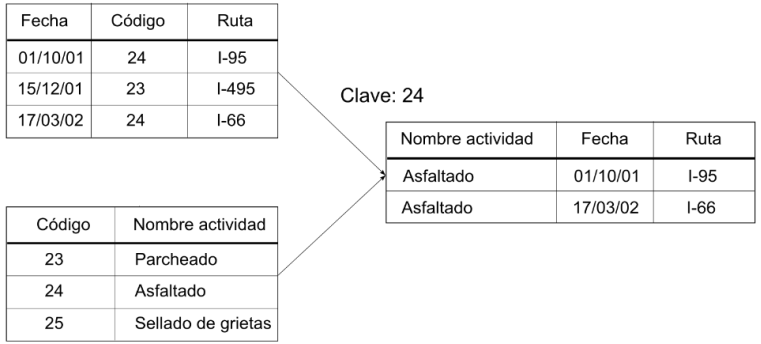
- **Vulnerabilidad a los fallos**
Al estar centralizada la información, el sistema se vuelve más vulnerable ante los fallos que puedan producirse.

Modelos de Bases de Datos

En función de la estructura utilizada para construir una BD, existen diversos modelos.
El modelo de la Base de Datos define un paradigma de almacenamiento, estableciendo cómo se estructuran los datos y las relaciones entre estos. Las distintas operaciones sobre la base de datos (eliminación o sustitución de datos, lectura de datos, etc.) vienen condicionadas por esta estructura. Algunos de los más habituales son los siguientes:

- **Bases de Datos Jerárquicas**:

→ **Bases de Datos Relacionales**:



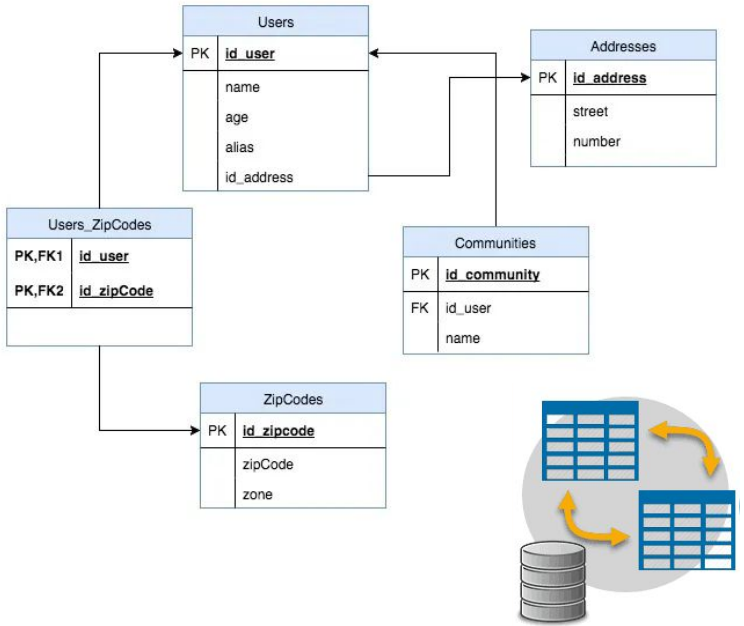
En esta etapa trabajaremos con Bases de Datos Relacionales, este modelo es el más utilizado en la actualidad.
Este esquema está basado en tablas. Las tablas contienen un número dado de registros (equivalentes a las filas en la tabla), así como campos (columnas), lo que da lugar a una correcta estructuración y un acceso eficiente.

Bases de Datos Relacionales

BD Relacionales

Son una colección de elementos de datos organizados en un conjunto de tablas.

La interfaz estándar de programa de usuario y aplicación a una base de datos relacional, es el Lenguaje de Consultas Estructuradas (SQL).
Las bases de datos relacionales utilizan un lenguaje de consulta estructurado para la manipulación de datos, estas se conforman por filas, columnas y registros y se almacenan por tablas. Para manipular los datos en SQL, se requiere primero determinar la estructura de estos, si se cambia la estructura de uno de los datos, puede perjudicar todo el sistema, ya que las tablas están relacionadas.



Bases de Datos No Relacionales

BD No Relacionales

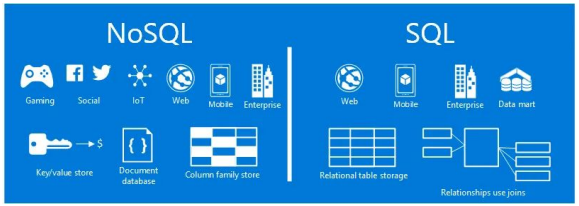
Están **diseñadas específicamente para modelos de datos específicos y tienen esquemas flexibles** para crear aplicaciones modernas. Son ampliamente reconocidas porque son fáciles de desarrollar, tanto en funcionalidad como en rendimiento a escala. Usan una variedad de modelos de datos que incluyen documentos, gráficos, clave-valor, en-memoria y búsqueda.

Las **bases de datos no relacionales (NoSQL)no tienen un identificador que sirva de relación entre un conjunto de datos y otros**. La información se organiza mediante documentos y es muy útil cuando no tenemos un esquema exacto de lo que se va a almacenar.

SQL vs. NoSQL

¿Cuándo utilizar SQL o NOSQL?

La elección entre SQL y NoSQL depende de varios factores, incluyendo la naturaleza de tus datos, requisitos de escalabilidad, y necesidades específicas de rendimiento.
SQL es ideal para aplicaciones que dependen de la integridad de los datos y relaciones complejas.
En contraste, NoSQL se adapta mejor a proyectos que requieren flexibilidad, escalabilidad y manejo eficiente de grandes volúmenes de datos no estructurados.



Es conveniente utilizar SQL cuando:
- El volumen de los datos no crece o crece poco a poco.
- Las necesidades del proceso se pueden asumir en un sólo servidor.
- No tenemos picos de uso del sistema fuera de lo previsto.

Se recomienda usar SQL para sistemas que requieren relaciones complejas y precisas entre los datos o aplicaciones que priorizan la integridad y seguridad de los datos, como sistemas bancarios y financieros.

Es conveniente utilizar NoSQL cuando:
- El volumen de los datos crece muy rápidamente en momentos puntuales.
- Las necesidades del proceso no se pueden prever.
- Tenemos picos de uso del sistema en múltiples ocasiones.

Se recomienda usar NoSQL para manejo de datos no estructurados o semiestructurados en grandes volúmenes, como logs de eventos o mensajes de redes sociales. Proyectos que requieren rapidez en el desarrollo y la capacidad de adaptarse rápidamente a cambios, gracias a la flexibilidad de su esquema.

Motores de Bases de Datos

Los motores de bases de datos son los artífices detrás del almacenamiento, recuperación y manipulación eficientes de información.
Su papel fundamental es gestionar la interacción entre las aplicaciones y los datos almacenados, proporcionando una infraestructura sólida para la gestión de información en entornos diversos, desde aplicaciones empresariales hasta plataformas web.
Algunos de los motores más utilizados por los desarrolladores en la actualidad son:

- Microsoft SQL Server
- MySQL
- SQLite
- Oracle Database
- ODBC
- PostgreSQL

MySQL fue diseñado para velocidad y confiabilidad, a expensas de la total adherencia al SQL estándar.

Las aplicaciones que usan una base de datos MySQL acceden a ella a través de un proceso de servidor en el que los programas se comunican con el servidor anfitrión a través de una comunicación interprocesada que retransmite las peticiones. Debido a que el proceso del servidor se encuentra entre la base de datos y otras aplicaciones, permite un mayor control sobre quién tiene acceso a la base de datos.

MySQL ha inspirado una gran cantidad de aplicaciones de terceros, herramientas y bibliotecas integradas que amplían su funcionalidad y ayudan a facilitar su trabajo. Algunas de las herramientas de terceros más utilizadas son phpMyAdmin, DBeaver y HeidiSQL.

Ventajas de utilizar MySQL

Popularidad y facilidad de uso: Abundante documentación sobre cómo instalar y administrar una BD MySQL, así como herramientas de terceros que simplifican el proceso de uso de la base de datos.
Seguridad: MySQL posee contraseña para usuarios. Además, admite la gestión de usuarios y permite conceder privilegios de acceso usuario por usuario.
Velocidad: Es una solución de BD excesivamente rápida.
Replicación: MySQL admite varios tipos diferentes de replicación, lo que ayuda a mejorar la fiabilidad, la disponibilidad y la tolerancia a las fallas.

PostgreSQL, también conocido como Postgres, fue creado con el objetivo de ser altamente extensible y cumplir con los estándares.
PostgreSQL es una base de datos relacional de objetos, lo que significa que aunque es principalmente una base de datos relacional también incluye características – como la herencia de tablas y la sobrecarga de funciones – que se asocian más a menudo con las bases de datos de objetos.

Postgres es capaz de manejar eficientemente múltiples tareas al mismo tiempo, una característica conocida como concurrencia. Lo logra sin bloqueos de lectura gracias a su implementación de Control de Concurrencia mediante Versiones Múltiples (MVCC), que asegura la atomicidad, consistencia, aislamiento y durabilidad de sus transacciones, también conocida como cumplimiento del ACID.

Ventajas de utilizar PostgreSQL

- **Conformidad con SQL**: De acuerdo con la documentación oficial de PostgreSQL, PostgreSQL soporta 160 de las 179 características requeridas para el cumplimiento completo del núcleo de SQL:2011.
- **Código abierto y orientado a la comunidad**: Un proyecto completamente de código abierto, el código fuente de PostgreSQL es desarrollado por una gran y dedicada comunidad.
- **Extensible**: Los usuarios pueden extender PostgreSQL programáticamente. Uno puede designar un archivo de código objeto, como una biblioteca compartida, y PostgreSQL lo cargará según sea necesario.

SQLite es un SGBDR (Sistema Gestor de Base de Datos Relacionales) autónomo, basado en archivos y de código abierto, conocido por su portabilidad, fiabilidad y fuerte rendimiento incluso en entornos con poca memoria.

El sitio web del proyecto SQLite lo describe como una base de datos «sin servidor». La mayoría de los motores de bases de datos relacionales se implementan como un proceso de servidor en el que los programas se comunican con el servidor anfitrión a través de una comunicación interprocesada que retransmite las peticiones. Con SQLite, sin embargo, cualquier proceso que acceda a la base de datos lee y escribe directamente en el disco de la base de datos. Esto simplifica el proceso de configuración de SQLite, ya que elimina cualquier necesidad de configurar un proceso de servidor. Asimismo, no es necesario configurar los programas que utilizarán la base de datos de SQLite: sólo necesitan acceder al disco.

Ventajas de utilizar SQLite

- **Tamaño reducido**: La biblioteca SQLite es muy ligera y es totalmente autónoma, lo que significa que no hay dependencias externas que deba instalar en su sistema para que SQLite funcione.
- **Facilidad de uso**: SQLite se describe a veces como una base de datos de «configuración cero», es decir, que está lista para su uso desde el principio.
- **Portátil**: Una base de datos SQLite completa se almacena en un solo archivo. Este archivo puede ubicarse en cualquier lugar de una jerarquía de directorios y puede compartirse a través de medios extraíbles o protocolo de transferencia de archivos.

Introducción a SQL

El Lenguaje de Consulta Estructurada (SQL) es un lenguaje de programación para almacenar y procesar información en una base de datos relacional.

Los sistemas gestores de bases de datos relacionales (SGBDR) almacenan varias tablas de bases de datos y utilizan este lenguaje para agregar, actualizar, eliminar, buscar y recuperar (CRUD - Create, Read, Update, Delete) información de la base de datos. También se puede usar SQL para mantener y optimizar el rendimiento de la base de datos.

Entre los sistemas de administración de bases de datos relacionales podemos mencionar: MS SQL Server, MySQL o PostgreSQL son ejemplos de sistemas de administración de bases de datos relacionales.

¿Por qué es importante SQL?

El Lenguaje de Consulta Estructurada (SQL) es un lenguaje de consulta popular que se usa con frecuencia en todos los tipos de aplicaciones. Los analistas y desarrolladores de datos aprenden y usan SQL porque se integra bien con los diferentes lenguajes de programación. Por ejemplo, Python, con librerías como psycopg2 o SQLAlchemy, permite una integración fluida con bases de datos SQL, lo que facilita la creación de aplicaciones de análisis de datos robustas y escalables. Esto lo convierte en una herramienta esencial para científicos de datos e ingenieros que trabajan con grandes volúmenes de información. Además, SQL es muy fácil de aprender, ya que en sus instrucciones se utilizan palabras clave comunes en inglés.

Introducción a SQL

El lenguaje SQL está compuesto por comandos, cláusulas, operadores y funciones de agregado. Estos elementos se combinan en las instrucciones para crear, actualizar y manipular las bases de datos.

Comandos

Existen dos tipos de comandos SQL:
- **DDL** que permiten crear y definir nuevas bases de datos, campos e índices.(CREATE, DROP, ALTER)
- **DML** que permiten generar consultas para ordenar, filtrar, extraer y actualizar datos de la base de datos. (SELECT, INSERT, UPDATE, DELETE)

Cláusulas

Las cláusulas son condiciones de modificación utilizadas para definir los datos que se desean seleccionar o manipular: **FROM**, **WHERE**, **GROUP BY**, **HAVING**, **ORDER BY**.

Operadores y Funciones

Los operadores son palabras claves reservadas o símbolos que se puede usar para realizar algunos cálculos lógicos y matemáticos: **AND, OR**, **NOT**, **>**, **<**, **=**.

Las funciones de agregado se usan dentro de una cláusula **SELECT** en grupos de registros para devolver un único valor que se aplica a un grupo de registros: **AVG**, **COUNT**, **SUM**.

2

Modelado de datos

Para poder plasmar o documentar propuestas de solución que involucran código usamos herramientas visuales como los **Diagramas de Flujo**.

Para entender la manera en la que se almacenarán los datos tenemos herramientas de modelado que nos permiten crear una vista unificada de los datos de una organización. 👉

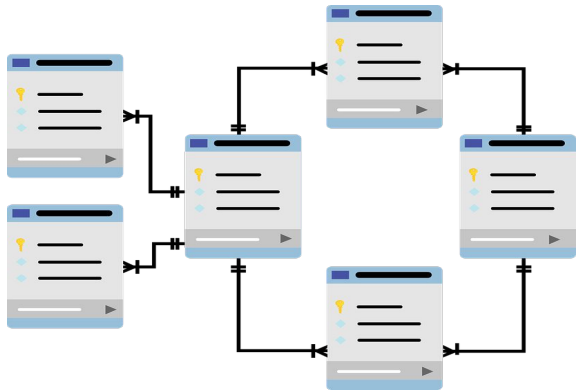
Modelado de Datos

El diseño de bases de datos es el proceso por el que se determina la organización de una base de datos, incluidos su estructura, contenido y las aplicaciones que se han de desarrollar.
Para ello necesitamos conocer sobre Modelos de Datos

¿Qué es un Modelo de Datos?

Representa un conjunto de conceptos para definir la estructura de la base de datos:

- Datos.
- Relaciones entre datos.
- Restricciones sobre datos y relaciones.
- Conjunto de operaciones para realizar consultas y actualizaciones de datos.



Modelado de Datos

Proceso de Diseño de Base de

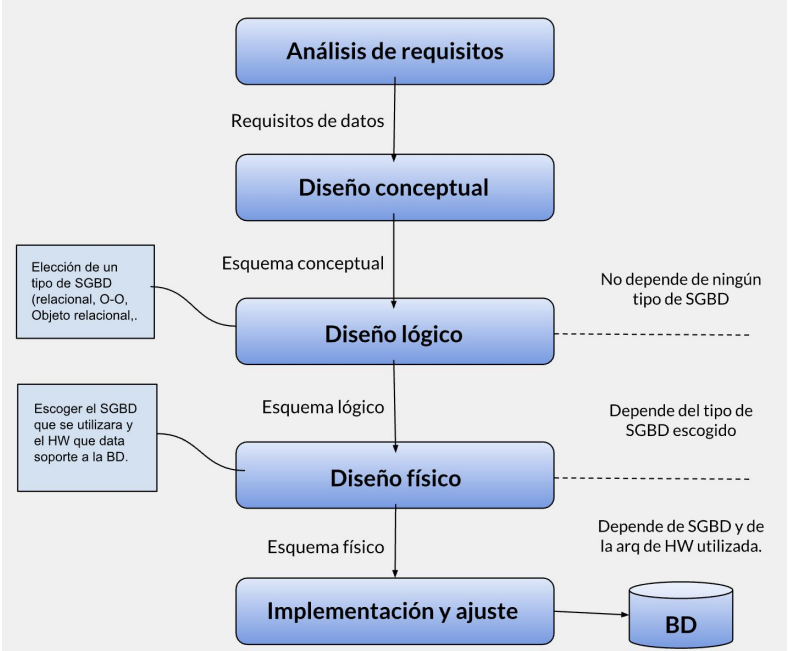
1) **Obtención y análisis de requisitos.**
El resultado de este paso será un conjunto de requisitos del usuario redactado de forma concisa.

2)**Esquema conceptual.**
Se crea un esquema conceptual para la base de datos mediante un modelo conceptual de datos de alto nivel.

3)**Esquema lógico.**
Selección del **SGBDR** que se utilizará para implementar la BD.

4)**Esquema físico.**
En base al SGBDR seleccionado, se planifican los recursos de HW que se necesitarán para dar soporte a la BD.

5)**Implementación y ajuste.**



Modelado de Datos

¿Cuáles son los elementos de un modelo?

→ **Entidad**

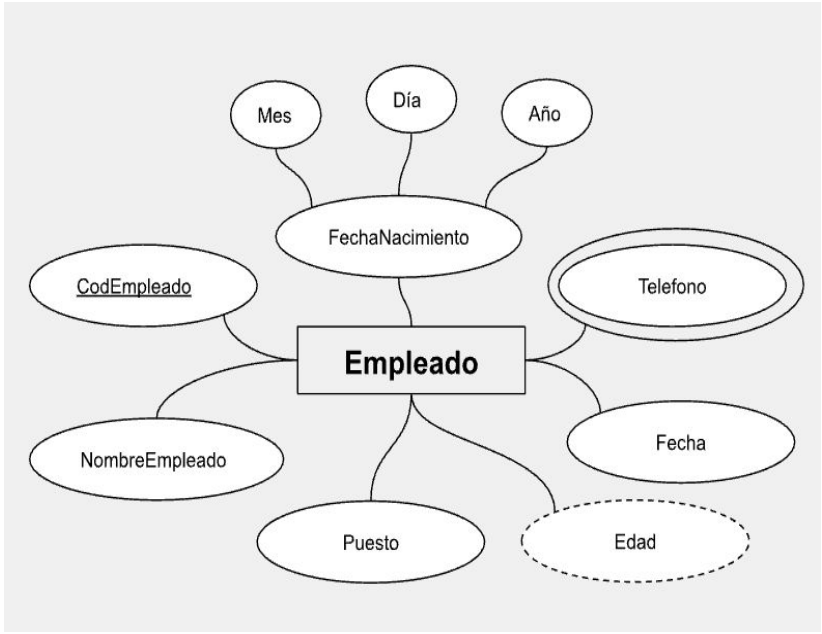
Una **entidad es un objeto del mundo real que puede distinguirse de otros objetos**.

Una entidad **puede ser un objeto con existencia física** (una persona, un automóvil, una casa, un empleado) o **un objeto con existencia conceptual** (una empresa, un puesto de trabajo, un curso universitario, una cuenta de cliente.

→ **Atributos**

Cada entidad tiene propiedades específicas llamadas atributos que la describen. Los atributos pueden ser:

- **Atributos compuestos o simples**
- **Atributos monovaluados o multivaluados**
- **Atributos almacenados o derivados**



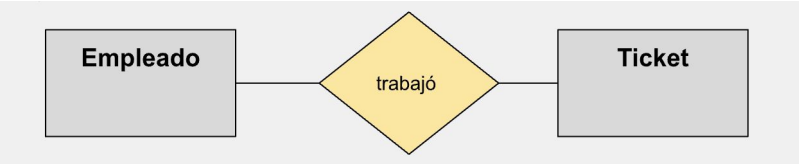
Modelado de Datos

¿Cuáles son los elementos de un modelo?

->Relación

Una relación es una **asociación entre dos o más entidades**.

Por ejemplo, puede que se tenga que modelar una relación en la cual un empleado haya trabajado en un ticket determinado.



En el conjunto de relaciones **“trabajó”, cada relación indica un trabajo efectuado por ese empleado**. Podemos observar que, puede que varios conjuntos de relaciones impliquen a los mismos conjuntos de entidades.

Modelado de Datos

¿Cuáles son los elementos de un modelo?

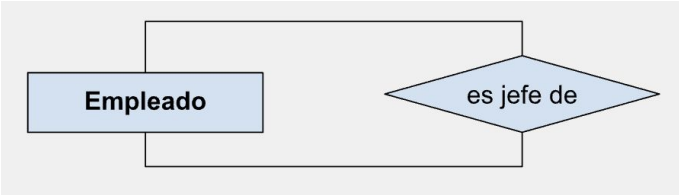
→**Relación**

- **Grado**

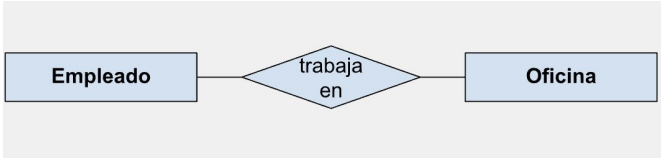
Se refiere al **número de entidades que participan en una relación**.
Los conjuntos de relaciones que involucran dos conjuntos de entidades se llaman relaciones binarias (o de grado dos). La mayoría de las relaciones en una base de datos es de este tipo.
Los conjuntos de relaciones pueden involucrar a más de dos conjuntos de entidades.

Nombre de la Relación	Grado
Unarias o Unitarias o de Reflexión	1
Binarias	2
Ternarias	3
N-arias	N

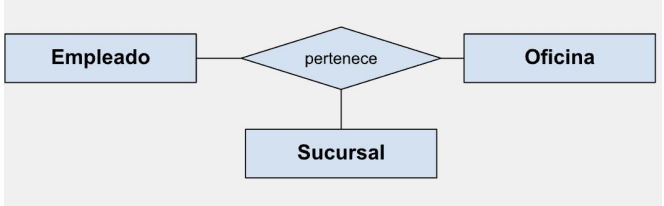
RELACIÓN UNARIA



RELACIÓN BINARIA



RELACIÓN TERNARIA



Modelado de Datos

¿Cuáles son los elementos de un modelo?

→**Relación**

- **Cardinalidad**

Dado un conjunto de relaciones en el que participan dos o más conjuntos de entidades, la cardinalidad de la correspondencia indica el número de entidades con las que puede estar relacionada una entidad dada. Las cardinalidades pueden ser:

- **Uno a Uno (1:1)**
- **Uno a Varios (1:N)**
- **Varios a Uno (N:1)**
- **Varios a Varios (N:M)**

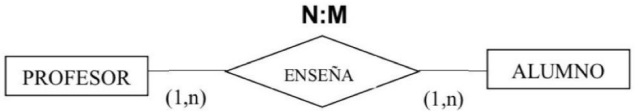
- **Atributos propios de una relación**

También las relaciones pueden tener atributos.
Son aquellos atributos cuyo valor sólo se puede obtener en la relación, puesto que dependen de todas las entidades que participan en la relación.

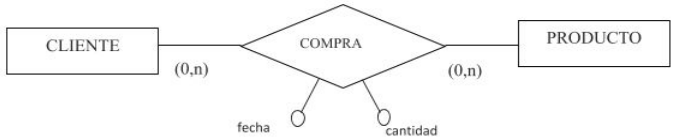
UNO A UNO (1:1)

UNO A VARIOS(1:N)

VARIOS A VARIOS (N:M)



ATRIBUTOS PROPIOS DE UNA RELACIÓN



Modelado de Datos

¿Cuáles son los elementos de un modelo?

→Clave

Una clave es un conjunto mínimo de atributos cuyos valores identifican de manera unívoca a cada entidad del conjunto.

→Clave Primaria (Primary Key)

Una clave primaria es una columna o conjunto de columnas en una tabla que identifica de forma única a cada registro en esa tabla. Es el primer paso hacia la creación de relaciones entre tablas en una base de datos relacional.

- **Características de una Clave Primaria:**
  - **Unicidad:** Cada valor en la columna o combinación de columnas debe ser único.
  - **Identidad:** Ninguna fila puede estar vacía en la columna o combinación de columnas designada como clave primaria.
  - **Indisponibilidad:** Los valores de la clave primaria no pueden ser nulos.

→Clave Primaria Compuesta (Composite Primary Key)

Una clave primaria compuesta es un conjunto de dos o más columnas que juntas identifican de manera única a cada registro en una tabla. Es especialmente útil cuando no hay una sola columna que pueda servir como identificador único..

Clave Foránea (Foreign Key)

Una clave foránea es una columna o un conjunto de columnas en una tabla cuyos valores corresponden a los valores de la clave primaria de otra tabla. Para poder añadir una fila con un valor de clave foránea específico, debe existir una fila en la tabla relacionada con el mismo valor de clave primaria.

Las relaciones de clave foránea se suelen hacer sobre la clave primaria de otra tabla. Esto permite vincular datos entre tablas y mantener la integridad referencial de la base de datos

Modelo de Entidad-Relación

El modelo de datos de entidad-relación (ER) se basa en una percepción de un mundo real que consiste en un conjunto de objetos básicos llamados entidades y de relaciones entre estos objetos.
Se desarrolló para facilitar el diseño de bases de datos permitiendo especificar un esquema empresarial. Este esquema representa la estructura lógica general de la base de datos.
Las *entidades*, son objetos que existen y que se distinguen de otros por sus características, por ejemplo:

PERSONAS

PROYECTOS

USUARIOS

MAQUINARIAS

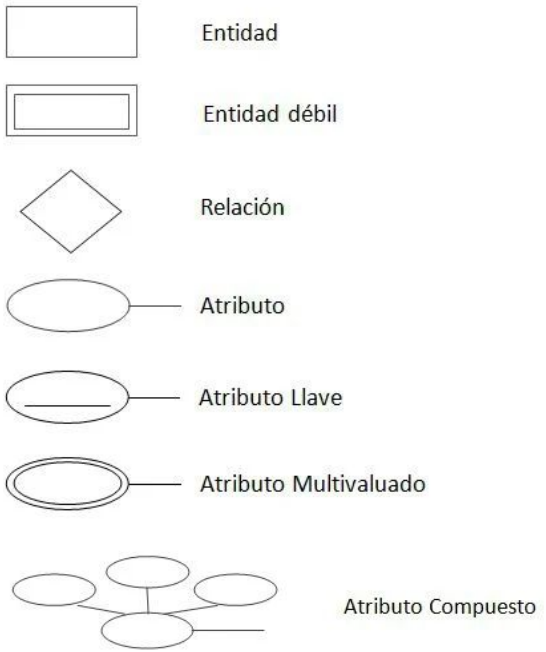
PRODUCTOS

Modelo de Entidad-Relación

Notación de Cheng

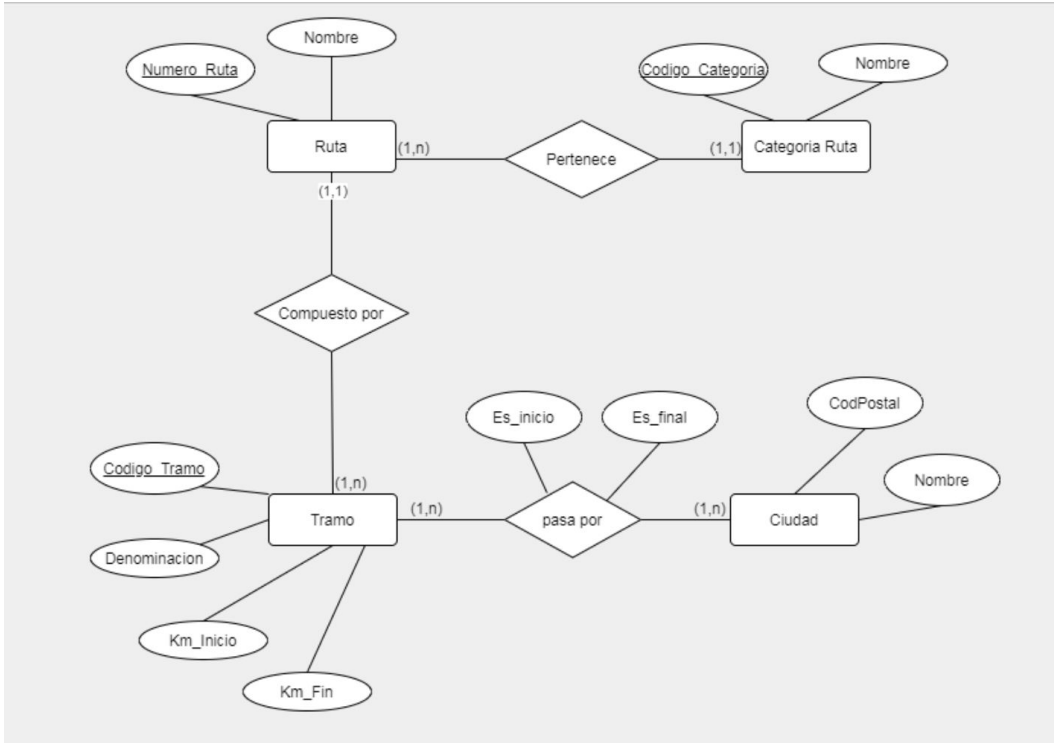
Para el Modelo de Entidad-Relación utilizaremos la notación de Chen:

- **Rectángulos**, que representan conjuntos de entidades.
- **Rectángulos dobles,** que representan conjuntos de entidades débiles.
- **Rombos**, que representan relaciones.
- **Líneas**, que unen atributos a conjuntos de entidades y conjuntos de entidades a conjuntos de relaciones.
- Líneas dobles, que indican participación total de una entidad en un conjunto de relaciones.
- **Elipses**, que representan atributos.
- **Elipses dobles**, que representan atributos multivalorados.
- **Elipses discontinuas**, que denotan atributos derivados.



Modelo de Entidad-Relación

Veamos un ejemplo:



Lo que vimos hoy 😉

Hasta acá pudimos ver una introducción a las bases de datos y conocer un poco sobre SQL. Además de ver cómo plasmar en Modelos de Entidad- Relación nuestras propuestas de bases de datos,

Recordá que vas a poder profundizar sobre este tema tanto en las clases de mentoría como en el material complementario.

Conclusión final

En esta clase, pudimos conocer un poco más sobre cómo funcionan las bases de datos y de qué manera podemos implementarlas. Conocimos la diferencia entre bases de datos relacionales, SQL, y NoSQL. Se exploraron los motores de bases de datos más comunes y sus características.

Por otro lado, se revisaron las técnicas de modelado de datos, enfocándose en la creación de un Modelo Entidad-Relación (MER), que es esencial para estructurar la información de manera lógica y eficiente.

Con esto se proporcionó una visión integral sobre cómo diseñar, implementar y gestionar bases de datos según las necesidades de diferentes tipos de aplicaciones.

¡Nos vemos En la próxima clase!