Una base de datos es una colección estructurada de datos que se almacena y se organiza de manera que permita un rápido acceso, gestión y actualización de la información en ella contenida.

1. Almacenamiento estructurado: Los datos se almacenan de forma organizada, generalmente en tablas compuestas por filas (registros) y columnas (campos), lo que facilita su gestión y consulta.

2. Integridad de datos: Las bases de datos incorporan mecanismos para garantizar la integridad y consistencia de los datos, evitando duplicados, datos erróneos o contradicciones.

3. Acceso concurrente: Varias aplicaciones y usuarios pueden acceder y modificar los datos de la base de datos de forma simultánea, gracias a los mecanismos de control de concurrencia implementados.

4. Lenguajes de consulta: Se utilizan lenguajes específicos, como SQL (Structured Query Language), para definir, manipular y consultar los datos de manera estructurada.

5. Gestión de transacciones: Las bases de datos soportan transacciones, que son unidades lógicas de trabajo que se ejecutan de forma completa o no se ejecutan en absoluto, manteniendo la integridad de los datos.

6. Seguridad y control de acceso: Las bases de datos incorporan mecanismos de autenticación y autorización para controlar el acceso a los datos y garantizar su seguridad.

Un gestor de base de datos, también conocido como sistema de gestión de bases de datos (SGBD o DBMS por sus siglas en inglés), es un software especializado diseñado para administrar y gestionar datos almacenados en bases de datos. Su función principal es servir como intermediario entre la base de datos y los usuarios o aplicaciones que necesitan acceder y modificar los datos.

Las principales funciones de un gestor de base de datos son:

1. DDL (Data Definition Language): Corresponde a la función de definición de datos. Incluye la creación, modificación y eliminación de estructuras de datos como tablas, índices, vistas, etc.

2. DML (Data Manipulation Language): Cubre la función de manipulación de datos. Incluye operaciones como insertar, actualizar, eliminar y consultar los datos almacenados en la base de datos.

3. DCL (Data Control Language): Se refiere al control de acceso. Gestiona los permisos y privilegios de acceso a los datos, asegurando la seguridad y confidencialidad de la información.

4. TCL (Transaction Control Language): Maneja las transacciones, permitiendo confirmar (COMMIT) o deshacer (ROLLBACK) conjuntos de operaciones de manera atómica.

Actualmente, existen dos tipos principales de bases de datos: relacionales y no relacionales. A continuación, te explico cada una de ellas:

Bases de Datos Relacionales (SQL):

- Estas bases de datos se basan en el modelo relacional, donde los datos se almacenan en tablas con filas y columnas.
- Utilizan el lenguaje SQL (Structured Query Language) para realizar operaciones como insertar, actualizar, eliminar y consultar datos.
- Ejemplos populares: MySQL, PostgreSQL, Oracle, Microsoft SQL Server.
- Son adecuadas para aplicaciones que requieren transacciones complejas, integridad de datos y consultas estructuradas.
- Siguen un esquema predefinido y tienen una estructura rígida.

Bases de Datos No Relacionales (NoSQL):

- A diferencia de las bases de datos relacionales, no utilizan el modelo de tablas con filas y columnas.
- Son más flexibles y escalables, lo que las hace adecuadas para grandes volúmenes de datos y aplicaciones distribuidas.
- Se dividen en varios tipos según su modelo de datos:

1. Clave-Valor: Almacenan datos como pares clave-valor. Ejemplos: Redis, Amazon DynamoDB.
2. Documentales: Almacenan datos en forma de documentos JSON o similares. Ejemplos: MongoDB, Couchbase.
3. Basadas en Grafos: Almacenan datos en forma de nodos y relaciones. Ejemplos: Neo4j, Amazon Neptune.
4. Bases de Datos en Memoria: Almacenan datos en la memoria principal para un acceso rápido. Ejemplos: Memcached, Redis.

- Son ideales para aplicaciones que requieren escalabilidad horizontal, esquemas flexibles y un alto rendimiento en operaciones de lectura/escritura.

Diseñar una base de datos relacional implica seguir ciertos pasos

1 Recopilar y analizar los requisitos

- Identifica las entidades principales que participarán en la base de datos (por ejemplo, clientes, productos, pedidos, etc.).
- Determina los atributos relevantes para cada entidad.
- Comprende las relaciones entre las diferentes entidades.

2 Crear el modelo Entidad-Relación (E-R)

- Representa gráficamente las entidades y sus atributos.
- Define las relaciones entre las entidades, ya sean uno a uno, uno a muchos o muchos a muchos.
- Asigna cardinalidades a las relaciones.

3 Normalización de datos

- Aplica las formas normales (1FN, 2FN, 3FN, etc.) para eliminar redundancias y dependencias no deseadas.
- Esto garantiza la integridad de los datos y evita problemas de actualización, inserción y eliminación.

4 Mapeo del modelo E-R al modelo relacional

- Convierte las entidades en tablas.
- Transforma los atributos en columnas de las tablas.
- Representa las relaciones como claves foráneas en las tablas relacionadas.

5 Definición de restricciones

- Establece las claves primarias para identificar inequívocamente cada fila en una tabla.
- Define las claves foráneas para mantener la integridad referencial.
- Agrega otras restricciones, como valores únicos, valores nulos permitidos, etc.

6 Diseño físico de la base de datos

- Decide el sistema de gestión de bases de datos (SGBD) que utilizarás.
- Crea el esquema de la base de datos utilizando el lenguaje de definición de datos (DDL) del SGBD.
- Implementa las tablas, restricciones y relaciones definidas en el modelo lógico.

7 Pruebas y refinamiento

- Realiza pruebas con datos de muestra para verificar la integridad y el rendimiento de la base de datos.
- Refina el diseño según sea necesario para abordar problemas o requisitos adicionales.

8 Documentación

- Documenta el diseño de la base de datos, incluyendo diagramas E-R,diccionario de datos, descripciones de tablas, relaciones y restricciones.
- Esto facilitará la comprensión y el mantenimiento futuro de la base de datos.

Caso de estudio:

Nos contrataron para desarrollar una aplicación web para una librería en línea. La librería vende libros, revistas y otros productos relacionados. Aquí están los requisitos principales:

- La librería necesita almacenar información sobre los clientes, incluyendo su nombre, dirección, correo electrónico y números de teléfono.
- Cada libro y revista tiene un título, autor(es), editorial, año de publicación, ISBN y precio.
- Los clientes pueden realizar pedidos, que pueden contener uno o más libros/revistas.
- Cada pedido tiene un número de pedido único, una fecha de pedido, una dirección de envío y un total.
- Los clientes pueden dejar reseñas para los libros y revistas que han comprado.
- La librería también vende otros productos, como bookmarks y bolsas de libros, con información como nombre, descripción y precio.

Guía de referencia:

1. **Recopilar y analizar los requisitos**
    - Entidades: Clientes, Libros, Revistas, Pedidos, Reseñas, OtrosProductos
    - Atributos y relaciones identificados en los requisitos
2. **Crear el modelo Entidad-Relación (E-R)**
    - Representar gráficamente las entidades, atributos y relaciones
    - Por ejemplo, un Cliente puede realizar muchos Pedidos, y un Pedido puede contener muchos Libros/Revistas
3. **Normalización de datos**

- Aplicar las formas normales para eliminar redundancias y dependencias no deseadas
- Por ejemplo, separar la información de autor y editorial en tablas separadas

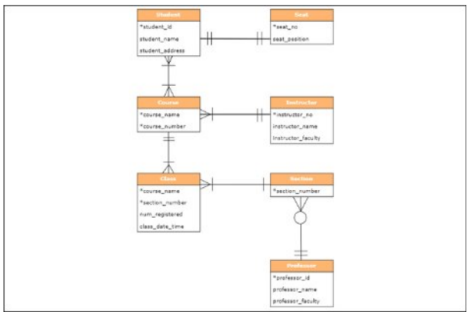
4. **Mapeo del modelo E-R al modelo relacional**
    - Convertir entidades en tablas (Clientes, Libros, Revistas, Pedidos, Reseñas, OtrosProductos)
    - Atributos se convierten en columnas
    - Relaciones se representan con claves foráneas
5. **Definición de restricciones**
    - Establecer claves primarias (por ejemplo, ID_Cliente, ISBN, ID_Pedido)
    - Definir claves foráneas (por ejemplo, Pedido referencia ID_Cliente)
    - Otras restricciones (por ejemplo, ISBN único, nombres obligatorios)
6. **Diseño físico de la base de datos**
    - Decidir el SGBD (por ejemplo, MySQL, PostgreSQL)
    - Crear el esquema de la base de datos utilizando DDL
    - Implementar tablas, restricciones y relaciones
7. **Pruebas y refinamiento**
    - Probar con datos de muestra
    - Refinar el diseño según sea necesario
8. **Documentación**
    - Diagramas E-R
    - Descripción de tablas, campos y relaciones
    - Restricciones y reglas de negocio

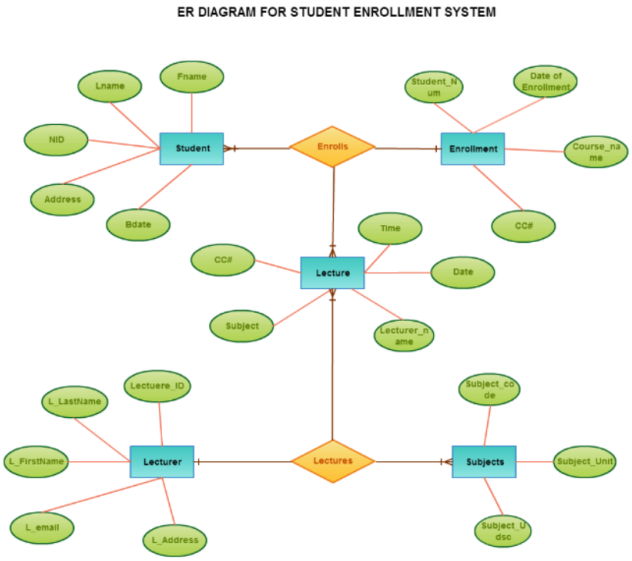
¿Qué es un diagrama ER?

Un diagrama de relación de entidad (ERD) es una representación visual de diferentes entidades dentro de un sistema y cómo se relacionan entre sí

¿Qué es un modelo ER?

Un modelo de entidad-relación representa la estructura de la base de datos con la ayuda de un diagrama. El modelado de ER es un proceso sistemático para diseñar una base de datos, ya que requiere que analice todos los requisitos de datos antes de implementar su base de datos.





Componentes principales y sus símbolos en los diagramas ER:

1

Entity Name

Entity

Person,place,object,event or concept about which data is to be maintained

Example: Car, Student

Jack

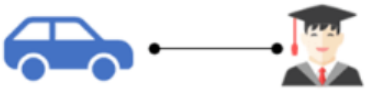
2

Attribute Name

Attribute

Property or characteristic of an entity

Example: Color of car Entity Name of Student Entity



Relation

3

Verb Phrase

Association between the instances of one or more entity types

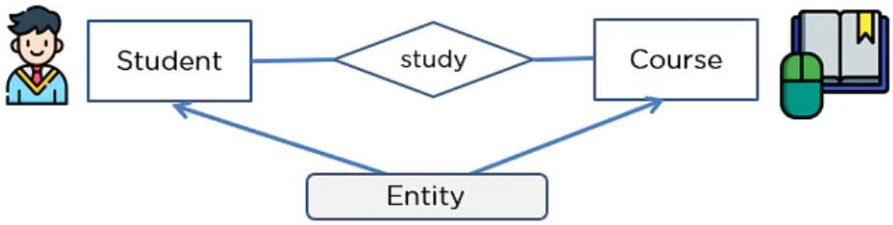
Example: Blue Car Belongs to Student Jack

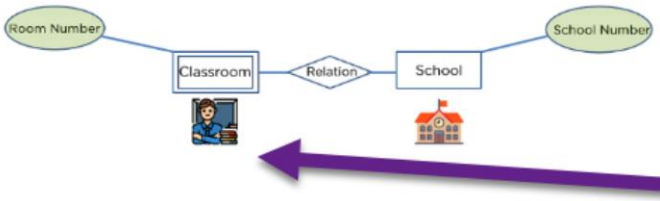
Entidades

Una entidad puede ser un componente vivo o no vivo.

Muestra una entidad como un rectángulo en un diagrama ER.

Por ejemplo, en un curso de estudio para estudiantes, tanto el estudiante como el curso son entidades.



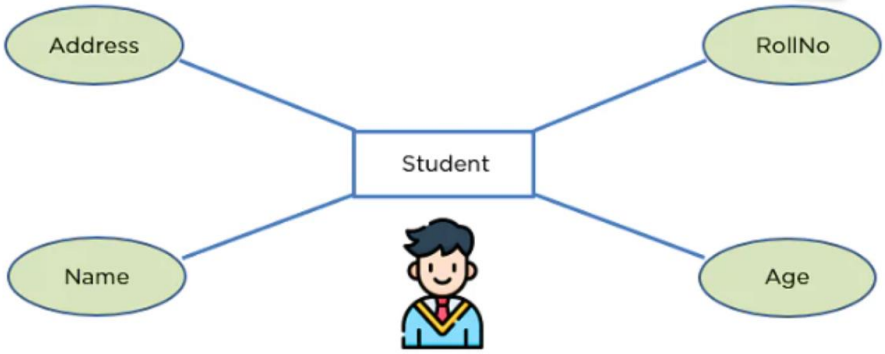


Una entidad débil es una entidad cuyos atributos no la identifican completamente, sino que sólo la identifican de forma parcial. Esta entidad debe participar en una interrelación que ayuda a identificarla.

Atributo 2

Un atributo exhibe las propiedades de una entidad.

Puede ilustrar un atributo con una forma ovalada en un diagrama ER.



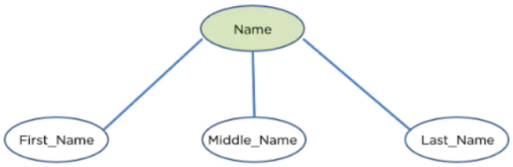
El atributo clave identifica de forma única una entidad de un conjunto de entidades.

Subraya el texto de un atributo clave.

Atributo compuesto

Un atributo que se compone de varios otros atributos se conoce como atributo compuesto.

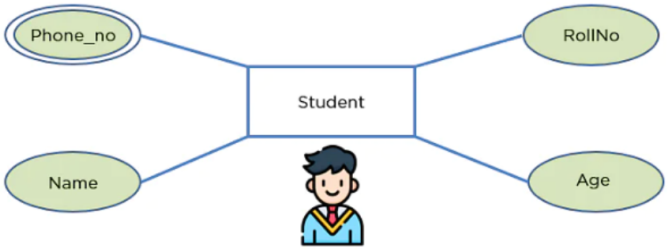
Un óvalo muestra el atributo compuesto, y el óvalo de atributo compuesto está más conectado con otros óvalos.



Atributo multivaluado

Algunos atributos pueden poseer más de un valor, esos atributos se denominan atributos multivaluados.

La forma de doble óvalo se utiliza para representar un atributo multivaluado.

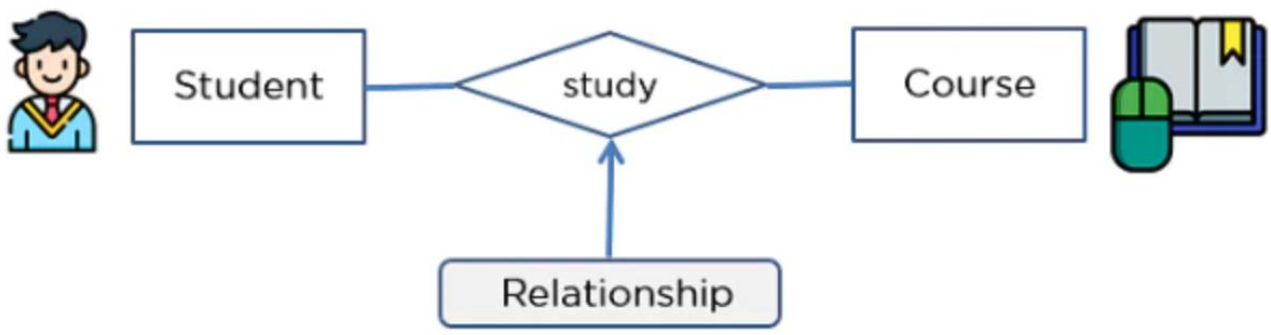


3 Relación

La forma de diamante muestra una relación en el diagrama ER.

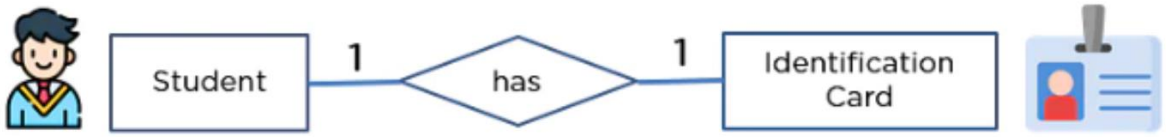
Representa la relación entre dos entidades.

En el siguiente ejemplo, tanto el estudiante como el curso son entidades, y el estudio es la relación entre ellos.



Cuando un solo elemento de una entidad se asocia con un solo elemento de otra entidad, se denomina relación uno a uno.

Por ejemplo, un estudiante tiene solo una tarjeta de identificación y se le da una tarjeta de identificación a una persona.



Relación de uno a muchos

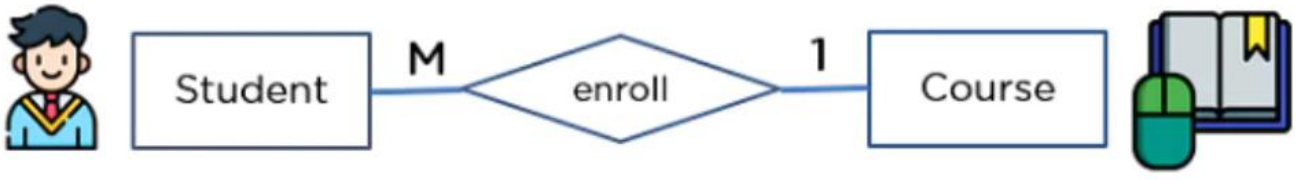
Cuando un solo elemento de una entidad está asociado con más de un elemento de otra entidad, se denomina relación de uno a muchos.

Por ejemplo, un cliente puede realizar muchos pedidos, pero muchos clientes no pueden realizar un pedido.

Relación de muchos a uno

Cuando más de un elemento de una entidad está relacionado con un solo elemento de otra entidad, se denomina relación de muchos a uno.

Por ejemplo, los estudiantes tienen que optar por un solo curso, pero un curso puede tener muchos estudiantes.



Relación de muchos a muchos

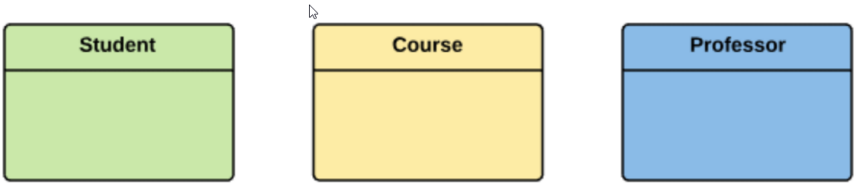
Cuando más de un elemento de una entidad está asociado con más de un elemento de otra entidad, esto se denomina relación de muchos a muchos.

Por ejemplo, puede asignar un empleado a muchos proyectos y un proyecto puede tener muchos empleados.

Paso 1) Identificación de la entidad

Tenemos tres entidades

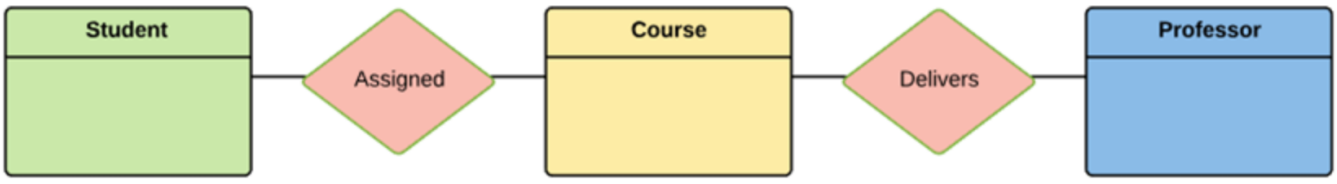
- Alumno
- Curso
- Profesor



Paso 2) Identificación de la relación

Tenemos las siguientes dos relaciones

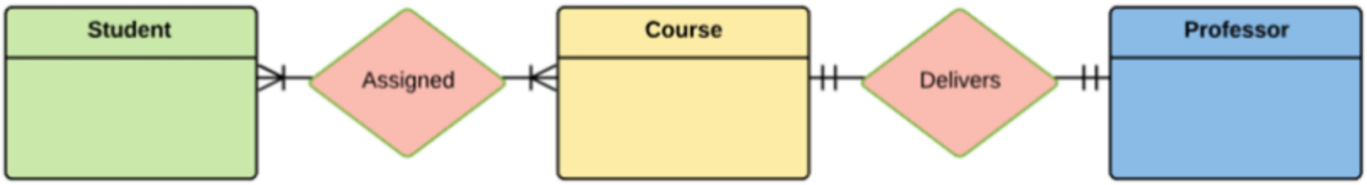
- El estudiante tiene **asignado** un curso.
- Profesor **imparte** un curso



Paso 3) Identificación de Cardinalidad

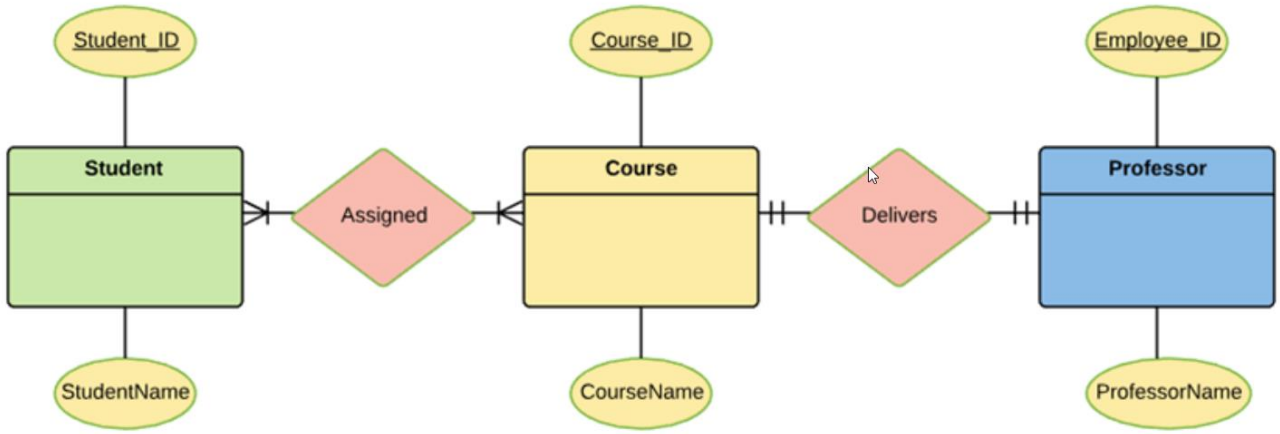
Para el enunciado del problema sabemos que,

- A un estudiante se le pueden asignar **varios** cursos
- Un profesor puede impartir sólo **un** curso.

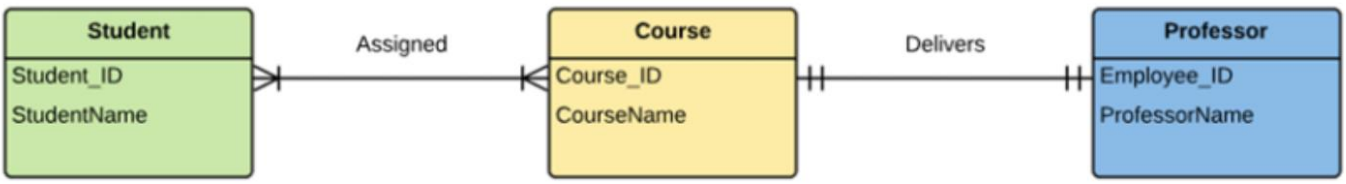


Paso 4) Identificar atributos

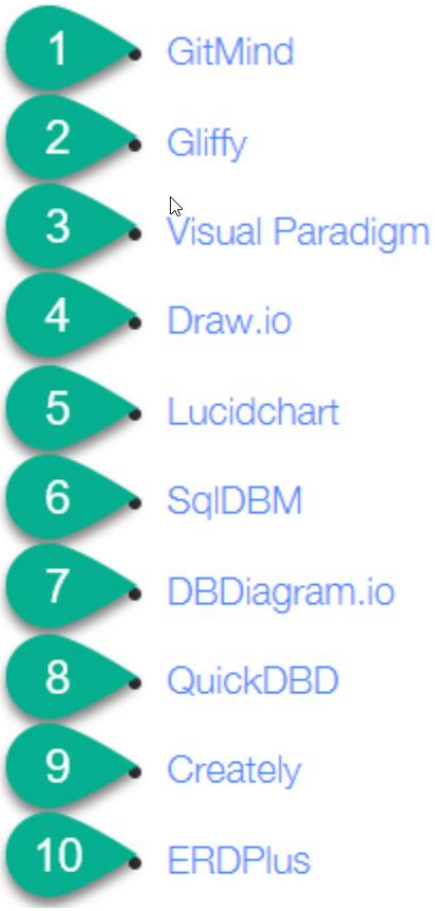
Entidad	Clave primaria	Atributo
Alumno	Identificación del Estudiante	Nombre del estudiante
Profesor	ID de empleado	ProfesorNombre
Curso	Curso_ID	Nombre del curso



Paso 5) Crear el Diagrama ER



Herramientas:



Hacer paso 1 y 2 del diseño con base en los siguientes casos de estudio:

Caso de estudio 2:

Debemos crear un sistema de gestión de inventario y ventas para una empresa de distribución de repuestos automotrices. Los requisitos principales son los siguientes:

1. La empresa necesita almacenar información sobre los proveedores, incluyendo nombre, dirección, contacto y productos que suministran.
2. Cada producto tiene un código único, descripción, precio de compra, precio de venta y categoría (frenos, motores, transmisión, etc.).
3. La empresa tiene múltiples almacenes ubicados en diferentes lugares, y cada almacén tiene una cantidad en stock de cada producto.
4. Los clientes pueden realizar pedidos, que pueden contener uno o más productos.
5. Cada pedido tiene un número de pedido único, una fecha de pedido, un total y la información del cliente (nombre, dirección, etc.).
6. Los empleados de la empresa pueden ser gerentes, vendedores o personal de almacén.
7. Se debe registrar el historial de movimientos de inventario, incluyendo entradas de nuevos productos, transferencias entre almacenes y ventas.

Caso de estudio 3:

Estamos creando un sistema de gestión para una universidad. Los requisitos principales son los siguientes:

1. La universidad necesita almacenar información sobre los estudiantes, incluyendo nombre, fecha de nacimiento, dirección, programa académico y año de ingreso.
2. Cada programa académico tiene un código único, nombre, departamento y un conjunto de cursos requeridos.
3. Cada curso tiene un código único, nombre, créditos, y puede tener uno o más profesores asignados.
4. Los profesores pueden dictar varios cursos, y cada profesor tiene información como nombre, departamento y títulos académicos.
5. Cada semestre, los estudiantes se inscriben en uno o más cursos, y se debe registrar la calificación obtenida en cada curso.
6. La universidad también tiene información sobre las aulas, incluyendo el número de aula, capacidad y ubicación.
7. Cada curso se imparte en un aula específica y en un horario determinado.