Modelamiento de Datos

¿Qué es el proceso de modelado de datos?

El proceso de modelado de datos sigue una secuencia de pasos que tiene que hacer repetidamente hasta crear un modelo de datos completo. Dentro de cualquier organización, varias partes interesadas se reúnen para crear una visión completa de los datos. Aunque los pasos varían según el tipo de modelado de datos, lo que sigue es una visión general.

Paso 1: identificar las entidades y sus propiedades

Identifique todas las entidades de su modelo de datos. Cada entidad tiene que ser lógicamente distinta de todas las demás y puede representar personas, lugares, cosas, conceptos o eventos. Cada entidad es distinta ya que tiene una o más propiedades únicas. Puede pensar en las entidades como sustantivos y en los atributos como adjetivos en su modelo de datos.

- Sustantivo (Entidad): PRODUCTO (o CLIENTE)
- Adjetivos (Atributos): Nombre, Precio, Color, Peso

Paso 2: identificar las relaciones entre entidades

Las relaciones entre las distintas entidades son el núcleo del modelado de datos. Las reglas empresariales definen inicialmente estas relaciones en un nivel conceptual. Puede pensar en las relaciones como los verbos de su modelo de datos.

Por ejemplo

- El vendedor vende muchos autos
- La sala de exhibiciones emplea a muchos vendedores.

Paso 3: identificar la técnica de modelado de datos

Después de entender conceptualmente sus entidades y sus relaciones, puede determinar la técnica de modelado de datos que mejor se adapte a su caso de uso. Por ejemplo, puede usar el modelado de datos relacional para los datos estructurados, pero el modelado de datos dimensional para los datos no estructurados.

Paso 4: optimizar y repetir

Puede optimizar aún más su modelo de datos para adaptarlo a sus necesidades tecnológicas y de rendimiento. Primero hay que determinar cómo se va a acceder a los datos y, a continuación, modelar los datos en la forma en que se va a acceder a ellos.

Por lo general, estos pasos se repiten a medida que la tecnología y los requisitos cambian con el tiempo.

Modelo Entidad-Relación

El Modelo Entidad-Relación (MER) es una herramienta fundamental en el diseño conceptual de bases de datos. Proporciona una representación clara y estructurada de cómo se organizan los datos en un sistema, permitiendo a los diseñadores abstraerse de detalles técnicos para enfocarse en la lógica del negocio.

Profundizaremos en los conceptos clave del MER, sus componentes principales (entidades, atributos y relaciones), y cómo utilizar Diagramas Entidad-Relación (DER) para modelar sistemas complejos. Además, resolveremos ejemplos prácticos y propondremos ejercicios para afianzar los conocimientos adquiridos.

Objetivos

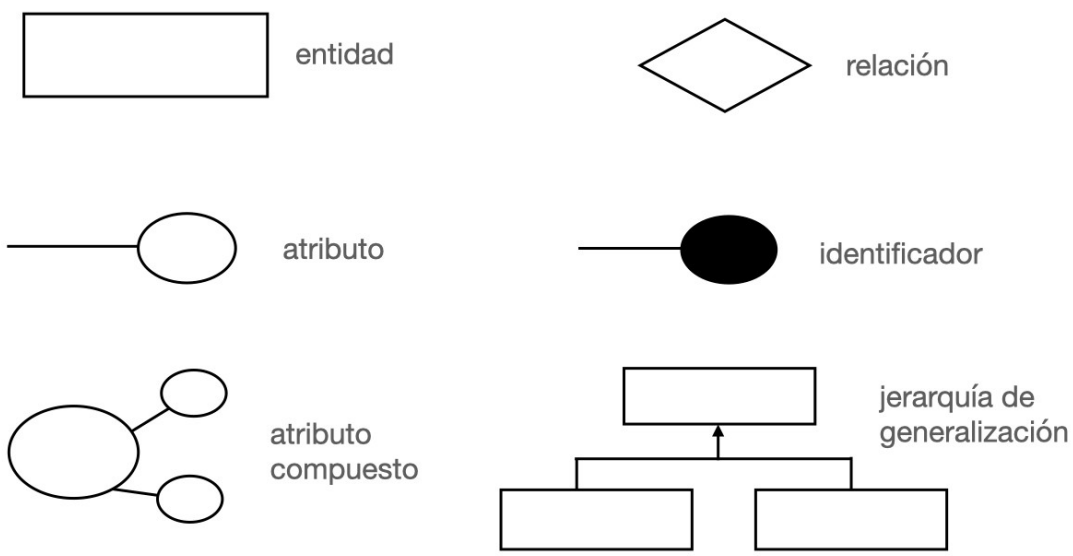
1. Comprender los fundamentos del Modelo Entidad-Relación.
2. Identificar y aplicar correctamente los elementos del MER: entidades, atributos, relaciones y cardinalidades.
3. Diseñar DER utilizando casos prácticos.
4. Resolver problemas mediante ejercicios de diseño conceptual.

Fundamentos del Modelo Entidad-Relación

¿Qué es el Modelo Entidad-Relación?

El MER es un modelo conceptual que describe las entidades (objetos o cosas sobre las que se desea almacenar información), sus propiedades (atributos) y las conexiones entre ellas (relaciones). Este modelo se utiliza como base para el diseño lógico de una base de datos antes de implementarla en un Sistema de Gestión de Bases de Datos (SGBD).

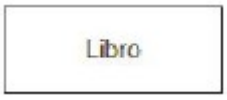
Componentes del MER



Identificador: O clave primaria

Entidad: Representa un conjunto de objetos o cosas que comparten características comunes. Por ejemplo:
- "Cliente"
- "Producto"
- "Empleado"

- **Entidades regulares**: son entidades por las cuales las ocurrencias tienen existencia propia. Gráficamente, una entidad se representa como un rectángulo.



- **Entidades débiles**: son entidades que dependen de la existencia de una concurrencia de la entidad regular, de forma que si desaparece una concurrencia

de la entidad regular, desaparecerán automáticamente todas las ocurrencias de la entidad débil dependientes. Una entidad débil se representa con un rectángulo concéntricos con el nombre de la entidad en el interior.

- Ejemplo :
  - Hotel – Habitación
  - Empleado – Dependiente
  - Factura – Detalle
  - Préstamo - Cuota

2. **Atributo**: Es una propiedad o característica de una entidad. Cada entidad tiene uno o más atributos que describen su estado. Por ejemplo:
- Para la entidad "Cliente": `ID_Cliente`, `Nombre`, `Dirección`, `Teléfono`.
- Para la entidad "Producto": `ID_Producto`, `Nombre`, `Precio`.

Los atributos pueden ser:

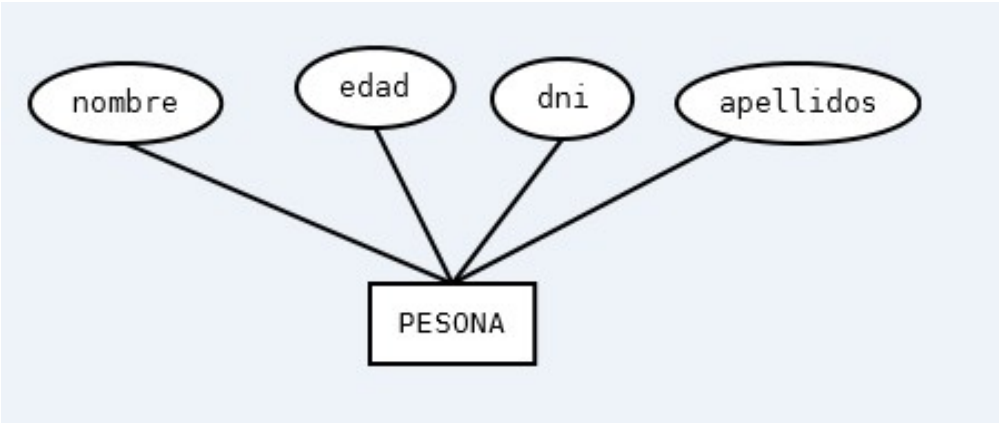
Atendiendo a los valores

Otra forma de clasificarlos es atendiendo al número de valores que puede tener cada atributo:

- **Monovaluados / Simples**

Un atributo monovaluado es aquel que tiene un solo valor por cada ocurrencia de la entidad a la que pertenece. Se representan mediante un círculo.

- Ejemplos: nombre, edad, dni, apellidos



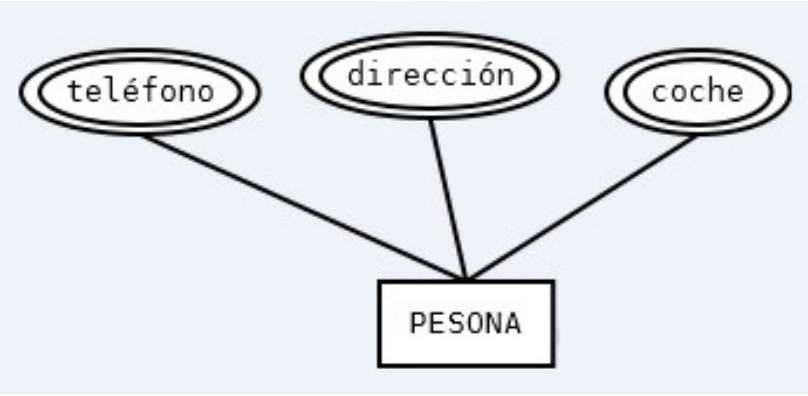
Compuesto

Se dividen en subpartes (ejemplo: `Dirección` → `Calle`, `Ciudad`, `País`).

Multivaluados

Por otra parte, un atributo multivaluado puede tener varios valores por cada ocurrencia de la entidad. Se representan de manera similar, pero en lugar de un círculo son dos, uno dentro de otro.

- Ejemplos: teléfono, dirección, coche



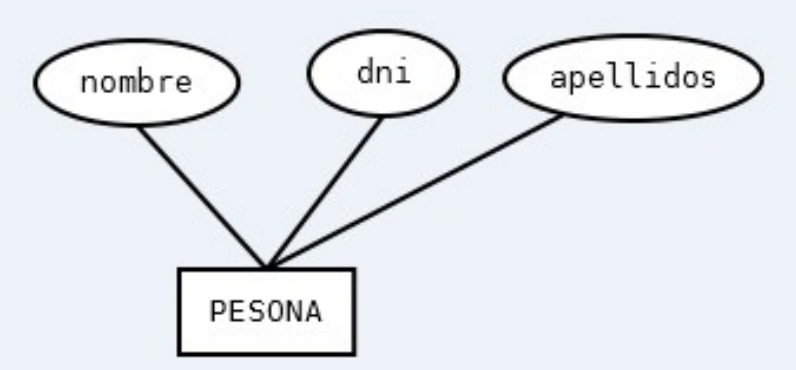
Atendiendo al origen

Se basa en el origen de los datos.

Almacenados

Son aquellos cuyos datos se almacenan directamente en la base de datos sin necesidad de realizar ningún trámite intermedio. Se representan mediante círculos.

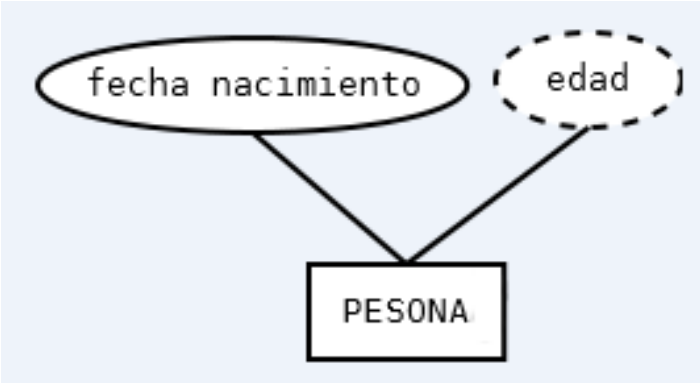
- Ejemplos: nombre, dni, apellidos



Derivados

Por contra, los atributos derivados son aquellos que son obtenidos a partir del valor de uno o varios atributos existentes en la misma o en otras entidades. Se representan mediante círculos discontinuos.

- Ejemplos: edad (a partir de la fecha de nacimiento)



Relación

Define cómo interactúan dos o más entidades entre sí. Por ejemplo:
- Un "Cliente" realiza una "Compra".
- Un "Profesor" imparte un "Curso".

Gráficamente, una relación se representa como un rombo conectado a las entidades involucradas.

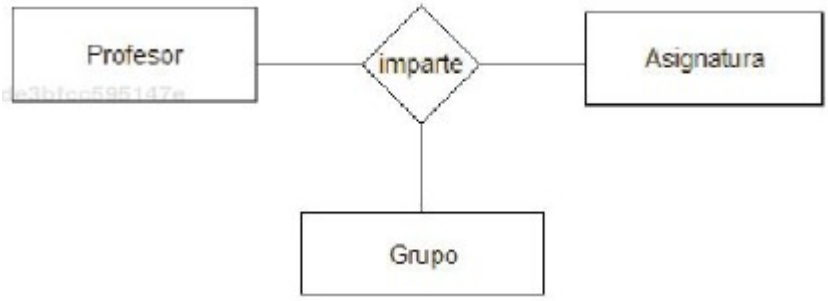
Características de una relación:

- Nombre: Toda relación debe tener un nombre único en el esquema.
- Grado: Numero de entidades que participan en una relación. Se clasifican en:

a) **Relaciones binarias o de grado 2**: Relacionan a dos entidades.

b) **Relaciones reflexivas o de grado 1**: Relacionan una entidad consigo misma.

c) **Relaciones ternarias, cuaternarias**,… o de grado 3,4,…: Relaciones tres, cuatro,… entidades respectivamente.



Cardinalidad

Las relaciones tienen **cardinalidade**s, que indican cuántas instancias de una entidad están relacionadas con otra. Las cardinalidades más comunes son:

- Uno a uno (1:1): Una instancia de una entidad se relaciona con una sola instancia de otra entidad. Por ejemplo, si tuviésemos una entidad con distintos chasis y otra con matrículas deberíamos de determinar que cada chasis solo puede tener una matrícula (y cada matrícula un chasis, ni más en ningún caso).

Cada Persona (entidad) está casado (relación) con una única Persona (entidad) y viceversa. Relación 1:1.

- Uno a muchos (1:N): Una instancia de una entidad se relaciona con varias instancias de otra entidad.

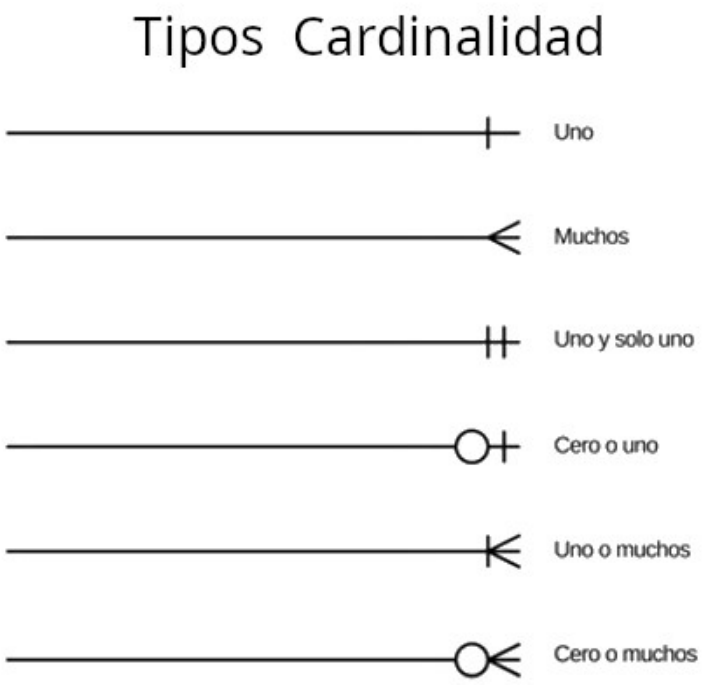
- Muchos a muchos (M:N): Varias instancias de una entidad se relacionan con varias instancias de otra entidad.

Cardinalidades y Restricciones

Las cardinalidades definen las reglas de participación de las entidades en una relación. Algunas restricciones importantes incluyen:

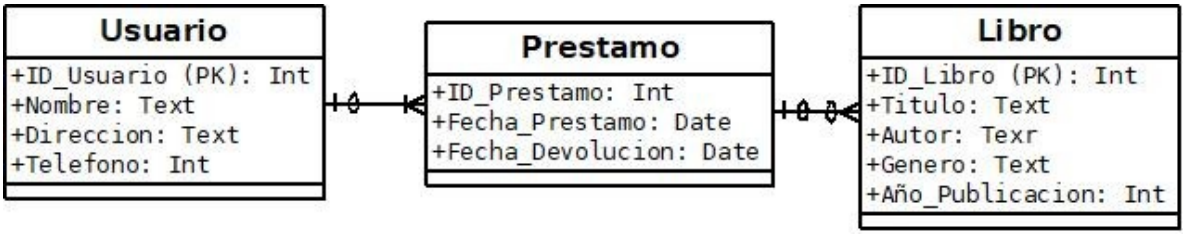
1. **Participación Total:** Todas las instancias de una entidad deben participar en la relación.
   - Ejemplo: Todo "Empleado" debe estar asignado a un "Departamento".
2. **Participación Parcial:** Solo algunas instancias de una entidad participan en la relación.
   - Ejemplo: No todos los "Clientes" han realizado una "Compra".

Gráficamente, la participación total se representa con una línea doble, mientras que la participación parcial se representa con una línea simple.

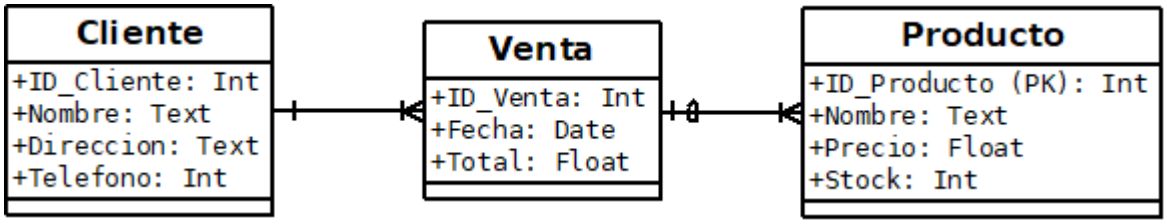

Tipos Cardinalidad
Uno
Muchos
Uno y solo uno
Cero o uno
Uno o muchos
Cero o muchos


Cliente
Llave Cliente_ID
Llave Nombre
Llave Apellido
Llave Calle
Llave Ciudad
Llave CP
Llave Teléfono
Pedido
PK Número_pedido
Llave Cliente_ID
Llave Nombre_cliente
Llave A_calle
Llave A_ciudad
Llave A_estado
Llave A_CP
Llave Fecha_envío
Producto
Llave Producto_ID
Llave Cantidad
Llave Tipo_producto

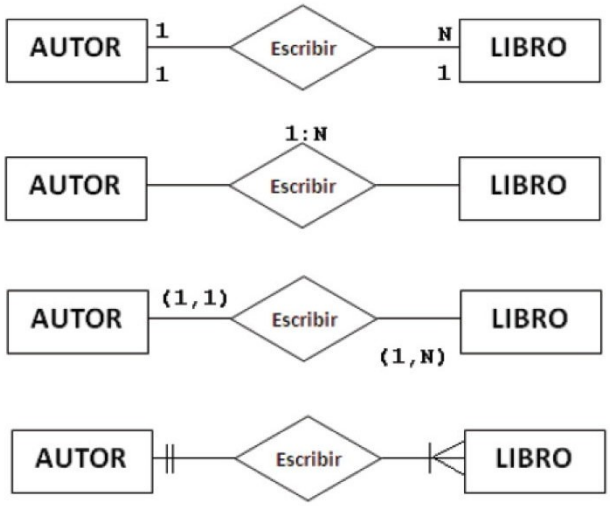
Ejemplo 1


Usuario
+ID_Usuario (PK): Int
+Nombre: Text
+Direccion: Text
+Telefono: Int
Prestamo
+ID_Prestamo: Int
+Fecha_Prestamo: Date
+Fecha_Devolucion: Date
Libro
+ID_Libro (PK): Int
+Titulo: Text
+Autor: Texr
+Genero: Text
+Año_Publicacion: Int

Ejemplo 2



Formas de representar Cardinalidades



El Modelo Entidad-Relación es una herramienta esencial para el diseño conceptual de bases de datos. Permite representar de manera clara y estructurada cómo se organizan y relacionan los datos en un sistema, facilitando la comunicación entre diseñadores, desarrolladores y usuarios finales.

Dominios

Cada atributo puede tener un conjunto de valores posibles. La descripción de los posibles valores de un atributo es lo que denominamos dominio

Ejemplos

El atributo NombredelDepartamento puede definirse como "el conjunto de cadenas con más de siete caracteres que representan los departamentos de Universidad".

Otros ejemplos de dominios son:

- Nombre: cadena de 10 caracteres
- Edad: número
- Fecha: date
- Peso: número con dos decimales
- Ciudad: cadena de 20 caracteres

Ejemplo Práctico 1

Sistema de Gestión de Biblioteca

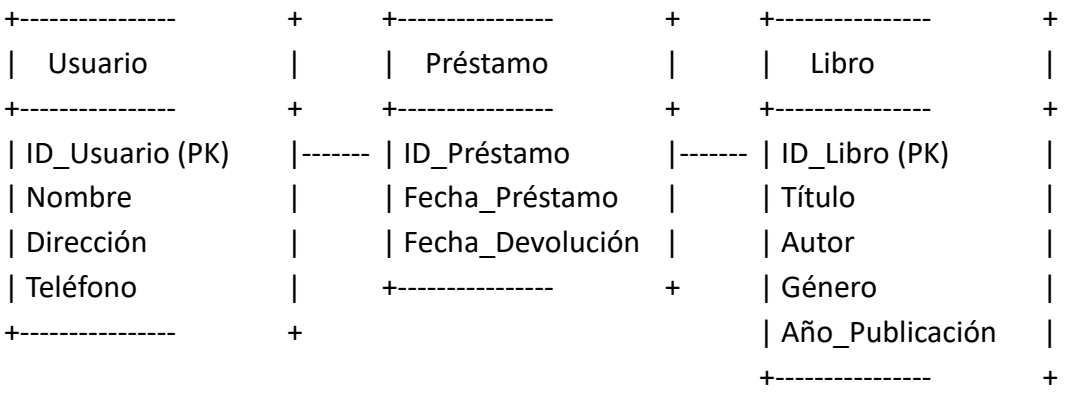
Descripción del Problema

Diseñar un MER para gestionar una biblioteca donde se almacene información sobre libros, usuarios y préstamos.

Entidades y Atributos
1. Libro:
  - Atributos: `ID_Libro` (clave primaria), `Título`, `Autor`, `Género`, `Año_Publicación`.
2. Usuario:
  - Atributos: `ID_Usuario` (clave primaria), `Nombre`, `Dirección`, `Teléfono`.
3. Préstamo:
  - Atributos: `Fecha_Préstamo`, `Fecha_Devolución`.

Relaciones
- Un *Usuario* puede tomar prestados varios *Libros* (1:N).
- Un *Libro* puede ser prestado a varios *Usuarios* en diferentes momentos (M:N).

Diagrama Final



Explicación:
La relación M:N entre "Usuario" y "Libro" se resuelve creando una nueva entidad llamada "Préstamo", que actúa como una tabla intermedia.

Ejercicio Práctico 2

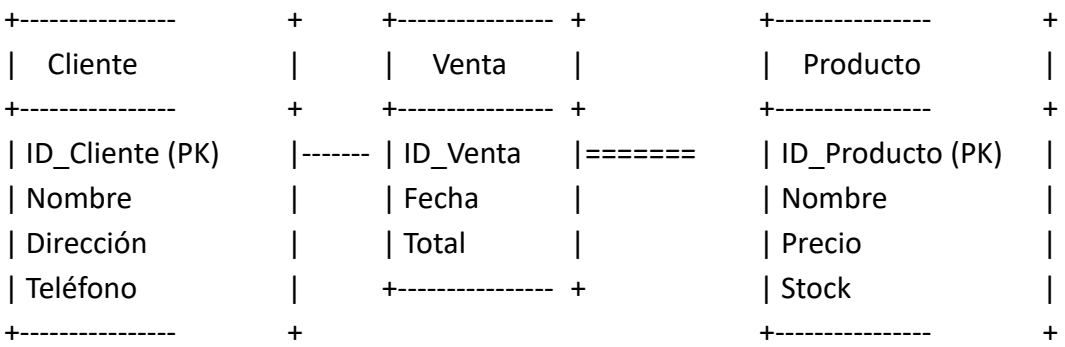
Problema 1: Sistema de Gestión de Tiendas

Diseñe un MER para un sistema de gestión de tiendas que cumpla con los siguientes requisitos:

1. La tienda gestiona *Productos* y *Clientes*.
   - Un *Producto* tiene un `*ID_Producto*`, `*Nombre*`, `*Precio*` y `*Stock*`.
   - Un *Cliente* tiene un `ID_Cliente`, `Nombre`, `Dirección` y `Teléfono`.
2. Los *Clientes* generan *Ventas*.
   - Una venta puede incluir varios productos.
   - Un producto puede estar en varias ventas.

Solución

Diagrama Final - Clases



Explicación:
- La relación M:N entre "Venta" y "Producto" se resuelve con una tabla intermedia llamada "Detalle_Venta", que podría incluir atributos como `Cantidad` y `Subtotal`.

Importante ver:

https://www.youtube.com/watch?v=URUkXTGQy1k

https://www.youtube.com/watch?v=Ki6HDNP9-Rk

https://www.youtube.com/watch?v=minaMCUHZno

Etapas en el diseño de bases de datos

Modelamiento Conceptual

El modelamiento conceptual es la fase inicial del diseño de una base de datos en la que se crea una representación abstracta y de alto nivel de la estructura de datos y las reglas del negocio, independiente de cualquier sistema gestor de bases de datos (DBMS) o consideración técnica de implementación.

Su objetivo principal es entender y representar los elementos clave del mundo real que son relevantes para una organización o sistema, así como las relaciones entre ellos, desde una perspectiva lógica y semántica.

Características del Modelamiento Conceptual

1. Independencia tecnológica: No depende de un SGBD específico (como MySQL, PostgreSQL, MS-SQL, Oracle, etc.) ni de decisiones de implementación (índices, tipos de datos, rendimiento).
2. Enfoque en el dominio del problema: Se centra en representar entidades, atributos, relaciones, restricciones y reglas del negocio tal como las entiende el usuario o el experto del dominio.
3. Uso de herramientas gráficas: Se representa comúnmente mediante diagramas como el modelo Entidad-Relación (E-R).
4. Comunicación entre actores: Sirve como herramienta de comunicación entre analistas, desarrolladores, usuarios finales y expertos del dominio para validar los requisitos de datos.
5. Base para el diseño lógico: El modelo conceptual es la base para derivar el modelo lógico, que ya se adapta a un tipo de modelo de datos específico (por ejemplo, relacional, orientado a objetos, etc.).

Elementos comunes en el modelamiento conceptual

- Entidades: Objetos del mundo real (por ejemplo: Cliente, Producto, Pedido).
- Atributos: Propiedades de las entidades (por ejemplo: nombre, fecha, precio).
- Relaciones: Asociaciones entre entidades (por ejemplo: un Cliente realiza un Pedido).

- Cardinalidades: Número de instancias relacionadas (por ejemplo: uno a muchos, muchos a muchos).
- Restricciones: Reglas de negocio (por ejemplo: un pedido debe tener al menos un producto).

Ejemplo sencillo

En una universidad:

- Entidades: Estudiante, Curso, Profesor
- Relaciones: Estudiante cursa Curso, Profesor imparte Curso
- Atributos: Estudiante (código, nombre), Curso (código, nombre, créditos)
- Cardinalidad: Un estudiante puede cursar muchos cursos; un curso puede tener muchos estudiantes → relación N:M

El modelo entidad-relación (MER) es un tipo de modelo conceptual para el diseño de bases de datos. Se utiliza para representar la realidad a través de entidades, atributos y relaciones, facilitando la comprensión y diseño de bases de datos.

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.

Características

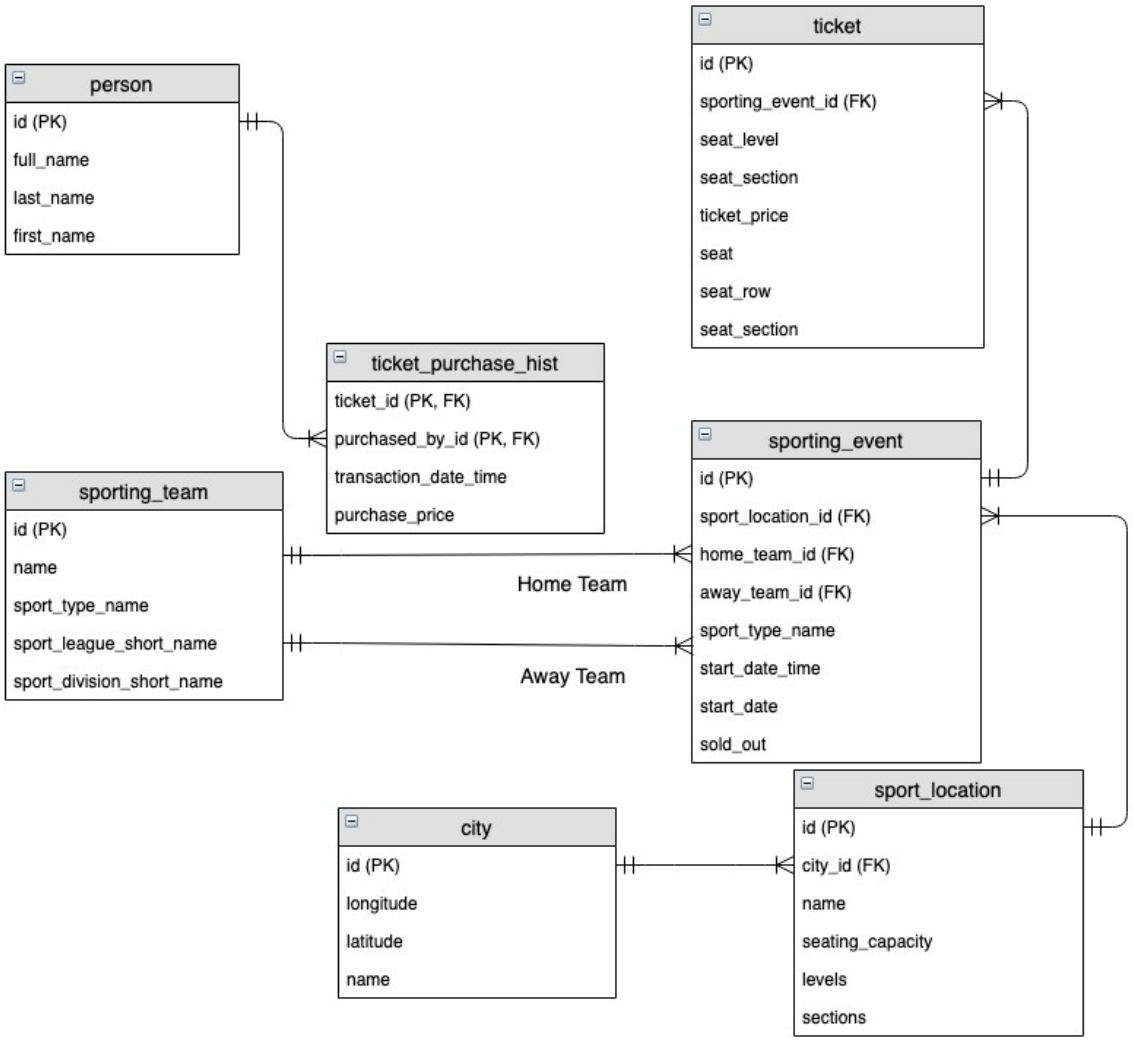
- Incluye todas las entidades y relaciones entre ellos.
- Todos los atributos para cada entidad están especificados.
- La clave o llave primaria para cada entidad está especificada.
- Se especifican las llaves externas o foraneas (claves que identifican la relación entre diferentes entidades).
- *La normalización ocurre en este nivel.*

Etapas

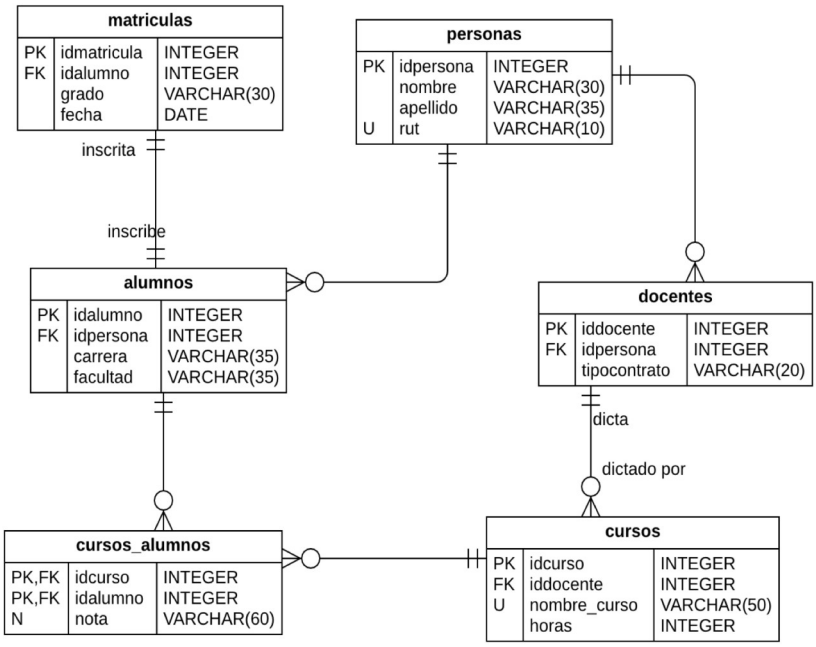
- Especifique llaves primarias para todas las entidades.
- Encuentra las relaciones entre diferentes entidades.
- Encuentra todos los atributos para cada entidad.
- Resuelva las relaciones de muchos a muchos.
- Normalización.

Nota:

https://dev.mysql.com/doc/refman/8.4/en/data-types.html


person
id (PK)
full_name
last_name
first_name
ticket
id (PK)
sporting_event_id (FK)
seat_level
seat_section
ticket_price
seat
seat_row
seat_section
ticket_purchase_hist
ticket_id (PK, FK)
purchased_by_id (PK, FK)
transaction_date_time
purchase_price
sporting_event
id (PK)
sport_location_id (FK)
home_team_id (FK)
away_team_id (FK)
sport_type_name
start_date_time
start_date
sold_out
sporting_team
id (PK)
name
sport_type_name
sport_league_short_name
sport_division_short_name
Home Team
Away Team
sport_location
id (PK)
city_id (FK)
name
seating_capacity
levels
sections
city
id (PK)
longitude
latitude
name

Modelamiento Físico



El modelo de datos físicos representa cómo se construirá el modelo en la base de datos. Un modelo de base de datos física 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.

Características

- 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.

Nota:

https://dev.mysql.com/doc/refman/8.4/en/data-types.html

Leer la siguiente documentación:

https://aws.amazon.com/es/compare/the-difference-between-logical-and-physical-data-model/

https://huridocs.org/community-resources/disenando-tu-modelo-de-datos-conceptual/

Cuadro comparativo detallado de los modelos de modelamiento de datos mas comunes

Aspecto	**Modelo Conceptual**	**Modelo Entidad-Relación (ER)**	**Modelo Lógico**	**Modelo Físico**
Objetivo	Capturar requisitos empresariales y conceptos clave.	Representar entidades, atributos y relaciones de forma visual.	Definir la estructura detallada sin dependencia técnica.	Implementar la base de datos en un sistema específico (DBMS).
Elementos principales	- Entidades - Relaciones (sin detalles técnicos) - Conceptos clave	- Entidades - Atributos - Relaciones - Cardinalidades	- Entidades detalladas - Atributos con tipos de datos - Claves primarias y foráneas - Normalización	- Tablas - Columnas con tipos específicos - Índices - Restricciones (PK, FK) - Optimizaciones (particiones)
Nivel de abstracción	Muy alto (orientado a negocio)	Alto (enfocado en estructura visual)	Medio (detalles técnicos sin implementación)	Bajo (implementación concreta)

Público objetivo	Stakeholders y equipos no técnicos	Analistas de datos, diseñadores y stakeholders	Arquitectos de datos y analistas técnicos	DBAs y desarrolladores
Detalle técnico	Sin atributos ni detalles técnicos	Incluye cardinalidades y atributos básicos	Tipos de datos genéricos, normalización	Tipos de datos específicos del DBMS, índices, SQL
Tipos de datos específicos del DBMS, índices, SQL	No depende	No depende	No depende	Totalmente dependiente
Ejemplo	Mapa conceptual con "Ventas" vinculado a "Clientes".	Diagrama ER con entidades "Cliente" y "Pedido", y su relación.	Tabla "Clientes" con campos: ID (INT), Nombre (VARCHAR).	Script SQL: `CREATE TABLE Clientes (ID INT PRIMARY KEY, ...);`
Fase de desarrollo	Fase inicial de requisitos	Fase inicial de diseño conceptual	Fase intermedia de diseño	Fase final de implementación
Herramientas comunes	Diagramas de flujo, UML (simplificado)	Lucidchart, Draw.io, ER/Studio	ERwin, PowerDesigner	SQL Server Management Studio, phpMyAdmin, pgAdmin, Oracle SQL Developer

Este cuadro refleja la progresión desde lo abstracto (negocio) hasta lo concreto (técnico), adaptándose a las necesidades de cada etapa del desarrollo.

Nomenclatura Común en Modelado Lógico y Físico

1. **PK – Primary Key (Clave Primaria)**
    1. Significado: Atributo o conjunto de atributos que identifica de forma única una fila en una tabla.
    2. Uso común: Se añade como sufijo o se indica en el diagrama.
    3. Ejemplos:
        1. *cliente_id PK*
        2. *id_cliente PK*
        3. *empleado_pk* (menos común, mejor usar id_empleado PK)

2. **FK – Foreign Key (Clave Foránea)**
    1. Significado: Atributo que hace referencia a la clave primaria de otra tabla, estableciendo una relación.
    2. Uso: Se indica junto al atributo.
    3. Ejemplos:
        1. *id_cliente FK* → referencia a cliente(*id_cliente*)
        2. *id_departamento FK* → referencia a departamento(*id_departamento*)
        3. En algunos diagramas, también se indica la tabla referenciada:
           id_cliente FK → cliente

3. **NN – Not Null (No Nulo)**
    1. Significado: El campo no puede tener valor nulo.
    2. Uso: Para indicar obligatoriedad del dato.
    3. Ejemplo:
        1. nombre NN → el nombre es obligatorio
        2. fecha_pedido NN

4. **UQ – Unique (Único)**
    1. Significado: El valor del campo debe ser único en la tabla (pero puede permitir un solo `NULL` en algunos SGBD).
    2. Uso: Para campos que no son clave primaria pero deben ser únicos.
    3. Ejemplo:
        1. *email UQ* → cada cliente tiene un correo único
        2. dni UQ

5. **AI – Auto Increment (Autoincrementable)**
    1. Significado: El valor del campo se genera automáticamente (usualmente entero secuencial).
    2. Uso común en campos tipo ID.
    3. Ejemplo:
        1. *id_cliente PK AI* → se genera automáticamente al insertar
        2. id_pedido PK AI
    4. Nota: En MySQL se llama *AUTO_INCREMENT,* en SQL Server *IDENTITY,* en PostgreSQL *SERIAL*.

6. **DF – Default (Valor por Defecto)**
    1. Significado: Valor que se asigna automáticamente si no se especifica otro.
    2. Uso: Se indica junto al valor.
    3. Ejemplo:
        1. estado DF 'activo'
        2. fecha_creacion DF CURRENT_DATE

7. **CK – Check Constraint (Restricción de verificación)**
    1. Significado: Restricción que limita los valores que puede tomar un campo.

2. Uso: Para reglas de negocio simples.
3. Ejemplo:
    1. *edad CK (edad >= 18)*
    2. *genero CK (genero IN ('M', 'F', 'O')) ó genero = 'M' OR genero = 'F' OR genero = 'O'*
        1. *IN significa "está dentro de esta lista de valores permitidos". En el ejemplo genero IN ('M', 'F', 'O'), asegura que el campo genero solo pueda contener 'M', 'F' o 'O', lo que garantiza integridad de datos y coherencia en la información almacenada.*

Ejemplo de tabla con nomenclatura (Modelado Lógico/Físico)

Tabla: cliente

Columna	**Tipo**	**Restricciones**	**Descripción**
id_cliente	INT	PK, AI	Identificador único del cliente
nombre	VARCHAR(50)	NN	Nombre del cliente
email	VARCHAR(80)	NN, UQ	Correo electrónico único
Telefono	VARCHAR(15)		Número de contacto
fecha_registro	DATE	DF CURRENT_DATE	Fecha de alta
estado	VARCHAR(10)	DF 'activo', CK (IN activo, inactivo)	Estado del cliente

Otras convenciones comunes de nomenclatura

Convención	**Descripción**	**Ejemplo**
id_ + nombre_singular	Para claves primarias	*id_cliente*, *id_producto*
nombre_tabla + _id	Alternativa común	*cliente_id, pedido_id*
Prefijos para tablas	Clasificar por módulo	*ven_pedido* (ventas), *rh_empleado* (recursos humanos)
Minúsculas y guiones bajos	Estilo recomendado (snake_case)	*fecha_nacimiento, precio_unitario*
Sin espacios ni acentos	Por compatibilidad	Usar *tipo_documento*, no `*tipo documento*`

Diferencia entre modelado lógico y físico en cuanto a nomenclatura

Aspecto	**Modelado Lógico**	**Modelado Físico**
PK, FK, NN, UQ	Sí, se indican	Sí, se detallan
Tipos de datos	Genéricos (ej. "texto", "número")	Específicos (ej. `VARCHAR(50)`, `INT`, `DATE`)
Índices	Rara vez	Sí, se definen (índices en FK, UQ, etc.)
Autoincremento	Opcional	Sí, se especifica
Nombres de tablas/columnas	Claros y semánticos	Con formato técnico y estándar (snake_case, sin espacios)

Resumen de abreviaturas comunes

Abreviatura	Significado
PK	Primary Key (Clave Primaria)
FK	Foreign Key (Clave Foránea)
NN	Not Null (No nulo)
UQ	Unique (Único)
AI	Auto Increment (Autoincrementable)
DF	Default (Valor por defecto)
CK	Check Constraint (Restricción de verificación)

Actividades Propuestas

Ejercicio 1: Sistema de Gestión de Clínicas
Diseñe el modelo conceptual, lógico y físico para una clínica que gestione pacientes, médicos y citas. Considere lo siguiente:

1. Un *Paciente* puede tener varias *Citas*.
2. Un *Médico* puede atender varias *Citas*.
3. Cada *Cita* tiene una fecha y hora específica.
4. Base de datos MySQL

Ejercicio 2: Sistema de Gestión de Universidades
Diseñe el modelo conceptual, lógico y físico para una universidad que gestione departamentos, profesores y cursos. Considere lo siguiente:

1. Un *Departamento* puede tener varios *Profesores*.
2. Un *Profesor* puede impartir varios *Cursos*.

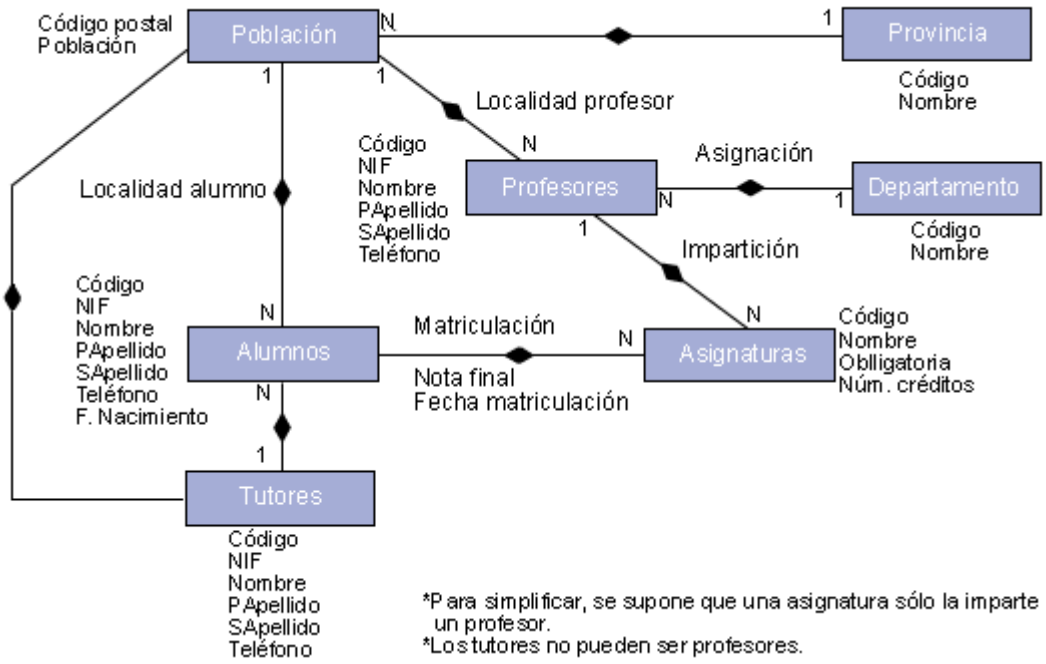
3. Un *Curso* puede ser impartido por varios *Profesores* en diferentes semestres.
4. Base de datos MySQL

Ejercicio 3. Taller de mecánica

Diseñe el modelo conceptual, lógico y físico para un taller de mecánica que gestione areas (lamina y pintura, mecánica general, alineación, balanceo, cambio de aceite), clientes, mecánicos, carros, fechas. Considere lo siguiente:

1. Un carro puede tener pasar por varias áreas (ej: Alineación y mecánica en general)
2. Una persona puede tener varios carros.
3. El carro debe de tener reporte del estado de entrada y salida.
4. Cada área del taller debe de tener un responsable.
5. Base de datos MySQL

Ejercicio 4: Construir las entidades y relaciones de acuerdo al siguiente diagrama



1. Crear las entidades (tablas) según el esquema, en una hoja de calculo preferiblemente.

2. Definir las llaves primarias
3. Definir las llaves foraneas
4. Están todas las llaves foraneas creadas?
5. Completar las entidades, agregando las llaves foraneas
6. Crear diagrama de clases, con las tablas completas.

Referencias

- Elmasri, R., & Navathe, S. B. (2016). *Fundamentals of Database Systems*. Pearson.
- Date, C. J. (2019). *Database Design and Relational Theory*. Apress.
- UML.org. (2023). *Unified Modeling Language Resources*.
- TuInstitutoOnline.com M.Donoso, G.García, P.Gargallo, A.Martínez. v. 2.0.2.1.0
- https://bookdown.org/paranedagarcia/database/modelamiento-de-datos.html

Videos

https://www.youtube.com/watch?v=Z0yLerU0g-Q

https://www.youtube.com/watch?v=TKuxYHb-Hvc

https://www.youtube.com/watch?v=jshi9VCTm7g