DISEÑO DE SISTEMAS

Juan Carlos Molina Lozano

Docente

OBJETIVOS DE LA CLASE

- Comprender los conceptos avanzados de los diagramas de clases.
- Identificar y aplicar relaciones avanzadas (herencia, asociación, composición y agregación).
- Utilizar estereotipos, restricciones y patrones en los diagramas de clases.
- Modelar sistemas complejos con UML y buenas prácticas.

PROPÓSITO DE LA CLASE

El estudiante debe desarrollar la capacidad de analizar, diseñar y representar estructuras complejas en UML mediante diagramas de clases, aplicando conceptos avanzados para mejorar la calidad del diseño del software.

METODOLOGÍA

- Exploración previa
- Aprendizaje basado en problemas
- Construcción colaborativa
- Discusión guiada

CONOCIMIENTOS PREVIOS

Qué es un diagrama de clases y cuál es su propósito en el desarrollo de software?

Qué tipo de relaciones conocen entre clases en UML?

Cuándo creen que es mejor usar herencia en lugar de composición?

DIAGRAMAS DE CLASES AVANZADOS

¿Qué es un diagrama de clases?

Es un diagrama estático en UML que muestra la estructura de un sistema, sus clases, atributos, métodos y las relaciones entre ellas.

¿Por qué es importante?

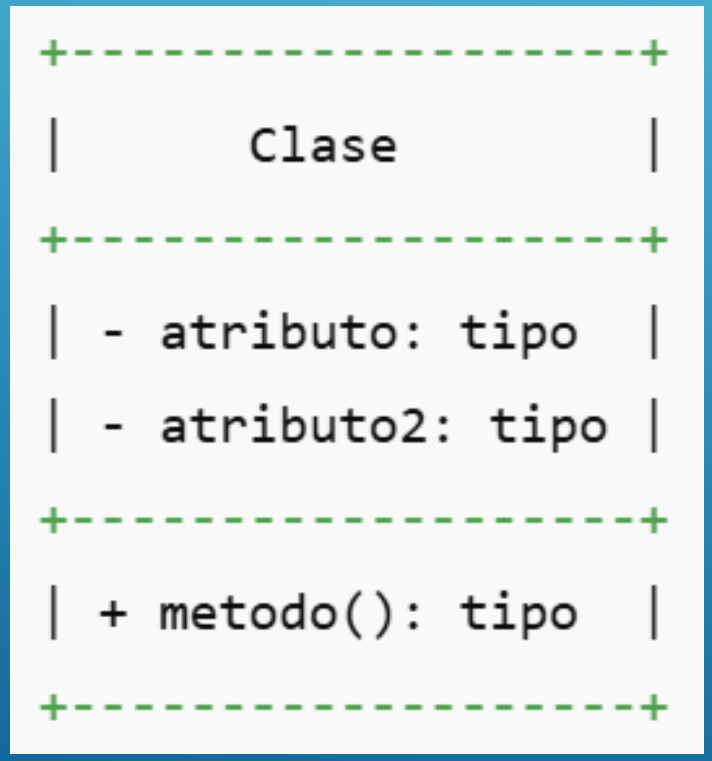
Permite visualizar el diseño del sistema, entender la arquitectura y facilitar la comunicación entre desarrolladores y stakeholders.

ELEMENTOS BÁSICOS DEL DIAGRAMA DE CLASES

1. Clases

Representan conceptos o entidades con atributos y métodos.

- Sintaxis:



Visibilidad:

+ público

- Privado

protegido

ELEMENTOS BÁSICOS DEL DIAGRAMA DE CLASES

2. Atributos y Métodos

- Tipos de atributos (simples, compuestos, derivados)
- Métodos: firmar, parámetros y visibilidad

3. Relaciones

- Asociación: vínculo estructural entre clases (línea simple)
- Multiplicidad: número de instancias relacionadas (1, 0..*, *, 1..n)
- Agregación: relación “todo-parte” débil (rombo vacío)
- Composición: relación “todo-parte” fuerte (rombo negro)
- Generalización (Herencia): relación padre-hijo (flecha con triángulo hueco)
- Dependencia: relación de uso temporal (línea punteada con flecha)

ELEMENTOS BÁSICOS DEL DIAGRAMA DE CLASES

EJEMPLOS

- Asociación: vínculo estructural entre clases (línea simple). Una clase conoce o tiene relación directa con otra.

Ejemplo: Alumno ———— Curso (Un alumno se inscribe en uno o más cursos.)

Representación UML: Alumno ---------- Curso

Línea simple entre clases.

Puede tener multiplicidad (1, 0..*, etc.) en ambos extremos.

- Multiplicidad: número de instancias relacionadas (1, 0..*, *, 1..n). Describe cuántas instancias de una entidad están relacionadas con otra.

Ejemplo: Supongamos una relación entre Profesor y Curso:

- Un Profesor puede enseñar entre 1 y 3 Cursos → multiplicidad del lado de Curso: 1..3
- Un Curso debe tener exactamente 1 Profesor → multiplicidad del lado de Profesor: 1

Esto se dibujaría así: Profesor "1" <----------> "1..3" Curso

ELEMENTOS BÁSICOS DEL DIAGRAMA DE CLASES

- Agregación: Relación “todo-parte” débil (rombo vacío).Una clase contiene a otra como parte, pero esa parte puede existir por sí sola.

Ejemplo: Departamento ◇———— Empleado

Un departamento tiene empleados, pero un empleado puede existir sin un departamento (por ejemplo, antes de ser asignado).

Representación UML: Departamento ◇---------- Empleado

Rombo vacío en el lado del "todo".

- Composición: relación “todo-parte” fuerte (rombo negro). Una clase contiene a otra como parte fundamental, y si el todo desaparece, la parte también.

Ejemplo: Libro ◆———— Capítulo

Un capítulo no puede existir sin un libro.

Representación UML: Libro ◆---------- Capítulo

Rombo negro en el lado del "todo".

ELEMENTOS BÁSICOS DEL DIAGRAMA DE CLASES

- Generalización (Herencia): relación padre-hijo (flecha con triángulo hueco). Una clase hija hereda atributos y métodos de la clase padre.

Ejemplo: Animal ◁———— Perro

Un perro es un animal.

Representación UML: Perro ----------|> Animal

Flecha con triángulo hueco apuntando hacia la superclase.

- Dependencia: Relación de uso temporal (línea punteada con flecha). Una clase usa a otra de manera temporal (por ejemplo, como parámetro en un método).

Ejemplo:Factura - - - - - > Cliente

La clase Factura depende de Cliente para obtener datos, pero no lo contiene ni lo hereda.

Representación UML: Factura - - - - - - - - > Cliente

Línea punteada con flecha.

Indica acoplamiento débil o uso.

ELEMENTOS BÁSICOS DEL DIAGRAMA DE CLASES

- Una clase implementa una interfaz (define su comportamiento).

Ejemplo: Imprimible ◁ - - - - Documento

Documento implementa la interfaz Imprimible.

Representación UML: Documento - - - - - |> Imprimible

Línea punteada con triángulo hueco apuntando hacia la interfaz.

ELEMENTOS BÁSICOS DEL DIAGRAMA DE CLASES

RELACIÓN	SÍMBOLO UML	EJEMPLO	SIGNIFICADO
Asociación	Línea simple	Alumno — Curso	Relación general
Agregación	Rombo vacío ◇	Departamento ◇— Empleado	Parte puede vivir sin el todo
Composición	Rombo negro ◆	Libro ◆— Capítulo	Parte muere con el todo
Herencia	Triángulo hueco ▷	Animal ◁— Perro	"Es un"
Dependencia	Línea punteada →	Factura - - - - > Cliente	Uso temporal
Interfaz	Línea punteada + ▷	Documento - - - -▷ Imprimible	Implementa comportamiento

CONCEPTOS AVANZADOS

1. Clases Abstractas e Interfaces

- Clase abstracta: no se instancia, puede tener métodos abstractos (sin implementación)
- Interfaces: solo definiciones de métodos (contratos)
- Notación UML:
- Clase abstracta en cursiva o estereotipo <<abstract>>
- Interfaces con estereotipo <<interface>>

2. Multiplicidad avanzada

- Rango, valores específicos
- Ejemplo: 1..*, 0..13.3

Multiplicidad	Significado
1	Uno y sólo uno
0..1	Cero o uno
N..M	Desde N hasta M
*	Cero o varios
0..*	Cero o varios
1..*	Uno o varios (al menos uno)

CONCEPTOS AVANZADOS

3. Asociaciones Reflexivas

- Una clase asociada consigo misma (ejemplo: una persona puede tener un "amigo" que es otra persona)

4. Clases anidadas (Nested Classes)

- Clase dentro de otra clase (uso común en ciertos lenguajes como Java)

5. Roles y Nombres de asociación

- Nombrar los roles en ambos extremos de la asociación para mayor claridad
- Ejemplo: Profesor enseña a Estudiante

IMPORTANCIA DE LOS DIAGRAMAS DE CLASES AVANZADOS

"Imaginemos que estamos diseñando el software de una plataforma como Amazon. ¿Cómo modelaríamos las relaciones entre productos, clientes, pagos y pedidos? ¿Bastará con clases básicas, o necesitaremos estructuras más avanzadas?“

💡 Explicación:

- Los diagramas de clases avanzados permiten modelar sistemas complejos con mayor precisión.
- Nos ayudan a representar la relación entre clases de manera eficiente y optimizada.
- Son fundamentales para el diseño de software escalable y bien estructurado.

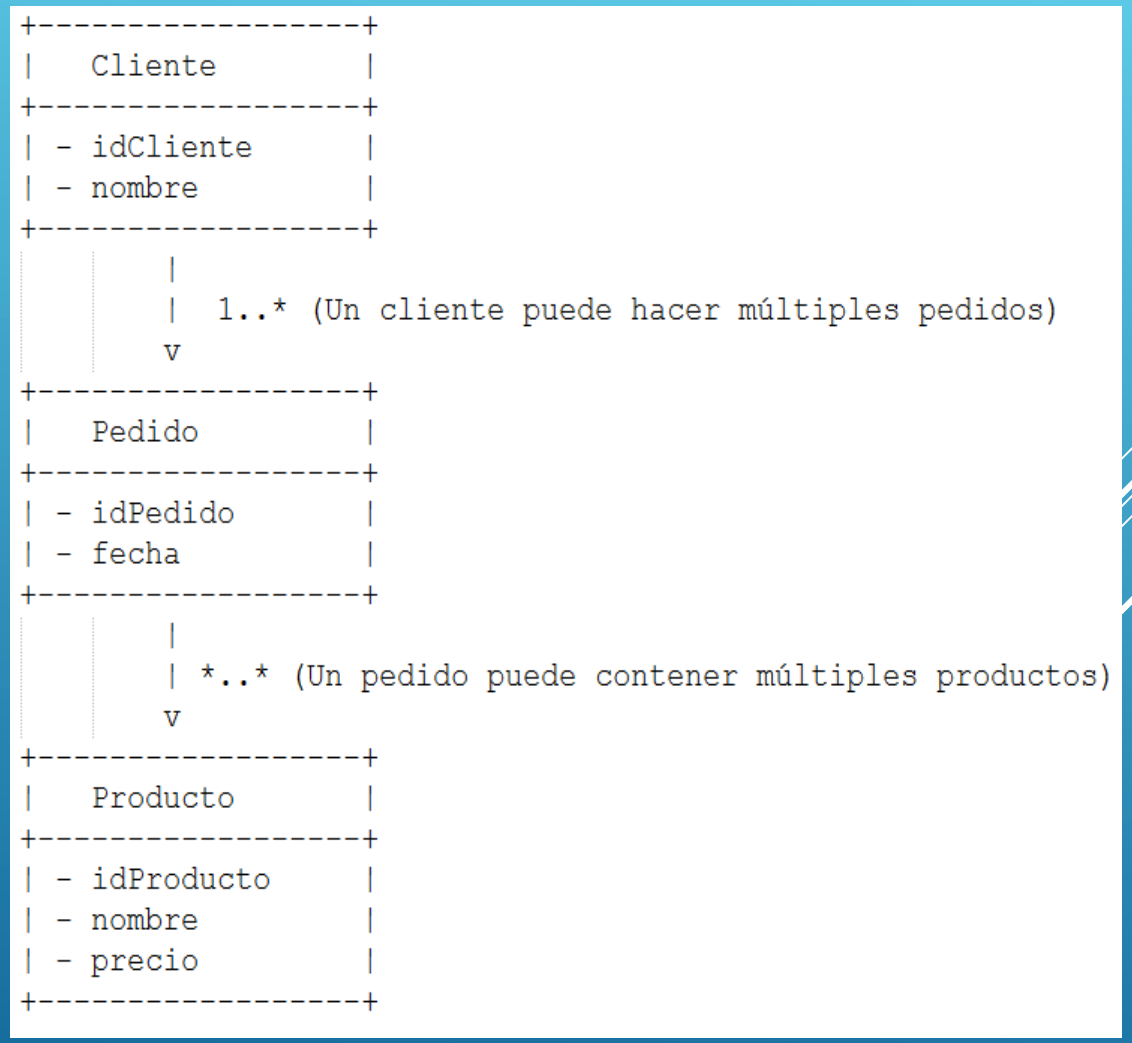
PRESENTACIÓN DEL PROBLEMA 1

Vamos a modelar un sistema de pedidos online, como si estuviéramos diseñando la estructura de Amazon o Mercado Libre. Tendremos clientes, pedidos, productos y diferentes métodos de pago.

- Cómo organizaríamos las clases para que el sistema sea flexible?
- Qué relaciones y estructuras nos permitirían hacer un diseño eficiente?
- Cómo aplicamos conceptos avanzados como herencia, composición y polimorfismo en UML?

PRESENTACIÓN DEL PROBLEMA

Diagrama de Clases Simple



Este diagrama representa un sistema básico donde:

- Un Cliente puede hacer múltiples Pedidos.
- Cada Pedido contiene varios Productos.

Limitaciones de este modelo simple:

- No distingue tipos de clientes (VIP, regular).
- No maneja métodos de pago.
- No permite aplicar descuentos o diferentes estados en el pedido.

PRESENTACIÓN DEL PROBLEMA

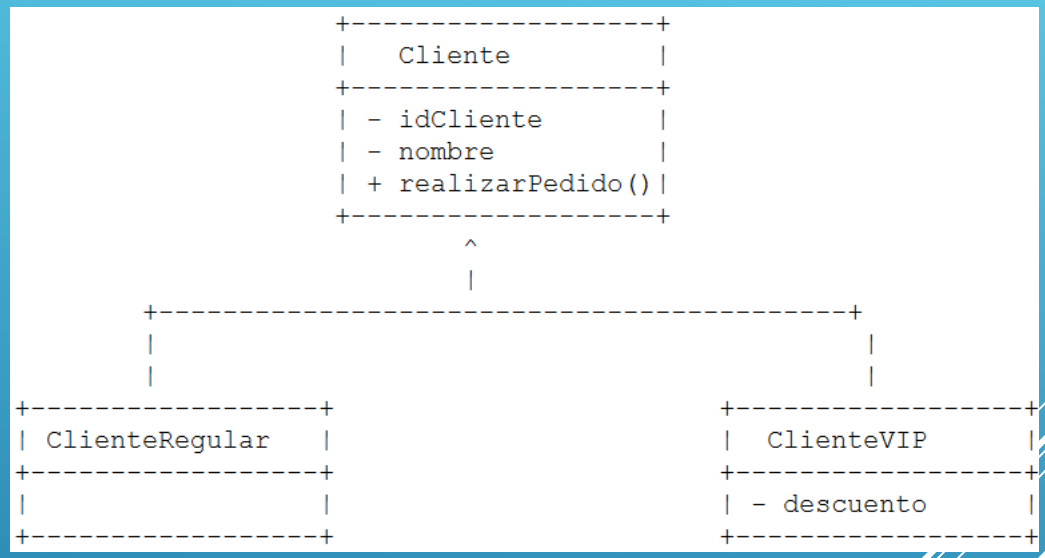
Diagrama de Clases Avanzado

Esta versión mejora el diseño agregando:

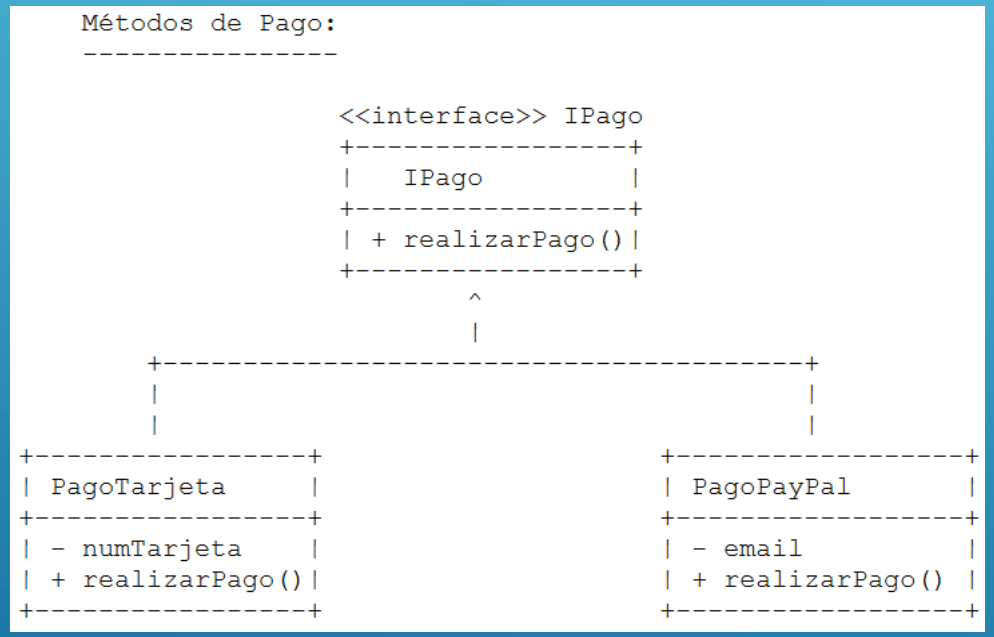
✔ Herencia: Clientes pueden ser Regulares o VIP.

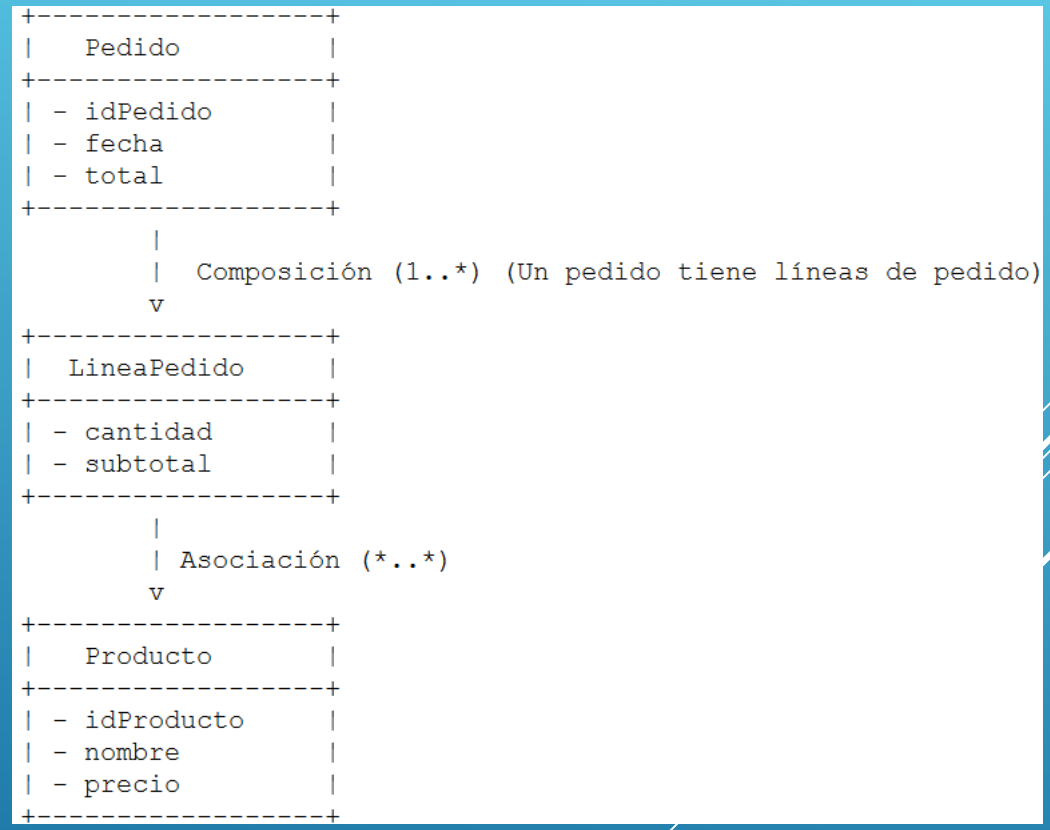
✔ Composición: Un pedido tiene Líneas de Pedido (si el pedido se elimina, sus líneas también).

✔ Uso de interfaces: Para manejar diferentes métodos de pago.



PRESENTACIÓN DEL PROBLEMA





📣 COMPARACIÓN ENTRE LOS DIAGRAMAS

Característica	Diagrama Simple	Diagrama Avanzado
Relación Cliente-Pedido	Sí	Sí
Relación Pedido-Producto	Sí	Sí
Diferenciación de Tipos de Cliente	❌ No	✅ Sí (Herencia)
Métodos de Pago	❌ No	✅ Sí (Interface)
Subdivisión de Pedido	❌ No	✅ Sí (Composición con LineaPedido)

PRESENTACIÓN DEL PROBLEMA 2

Diagrama de Clases Simple

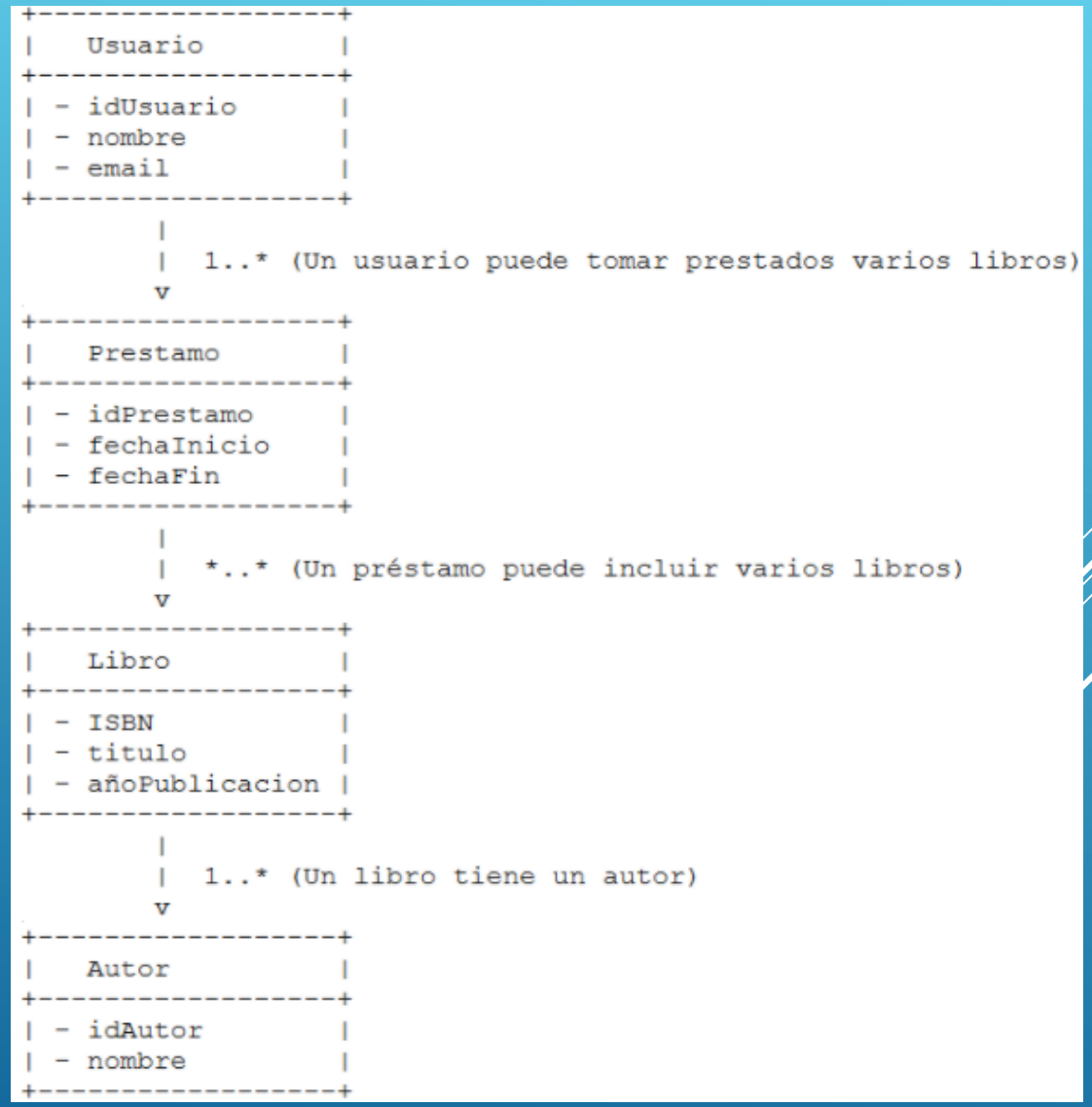
Este modelo básico representa:

✔ Un Usuario que puede tomar prestados Libros.

✔ Cada Libro tiene un Autor y está identificado por su ISBN.

Limitaciones del modelo simple:

- No distingue entre tipos de usuarios (estudiantes, profesores, visitantes).
- No maneja disponibilidad de libros.
- No incluye multas ni renovaciones.



PREGUNTAS