INFORME DEL PROYECTO: SISTEMA DE INVENTARIO DE TECNOLOGÍA

Nombre del Proyecto: Sistema de Inventario de Tecnología

Nombre del Grupo: Team 6

Luis Miguel David Campo

Juan Miguel Hernández Delgado

Jhonny Alejandro Gálvez

Asignatura: Bases de Datos

Docente: Froncisco Alexander Rojas

Fecha: Mayo de 2025

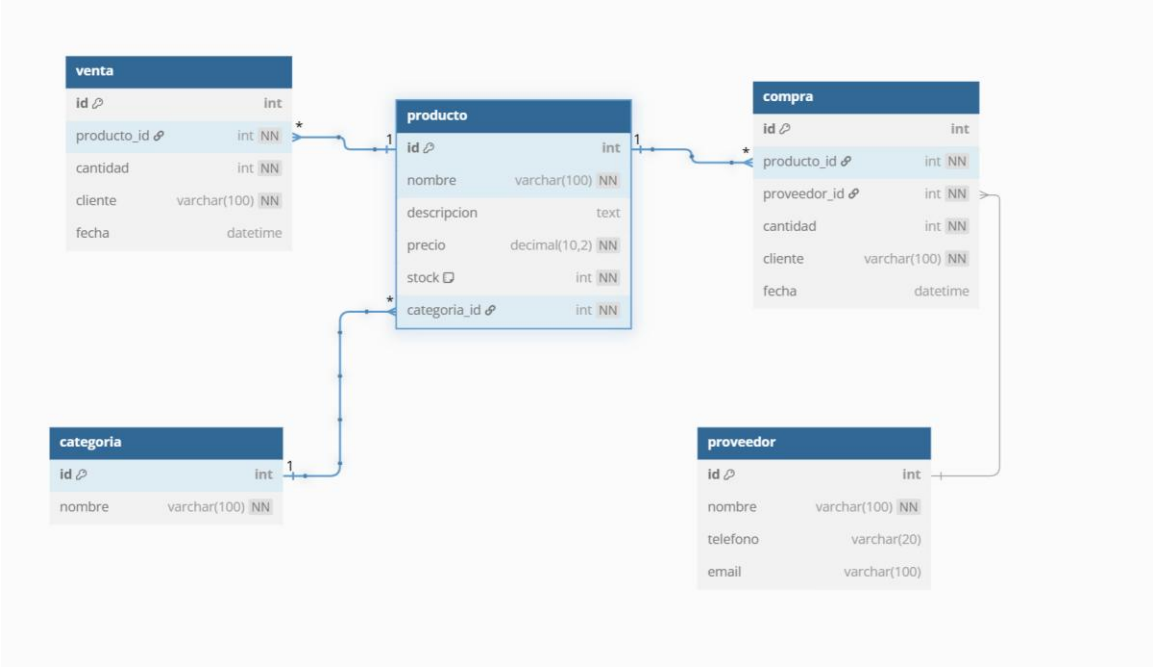
Idea General

El proyecto consiste en el desarrollo de un sistema web para la gestión de inventario de productos tecnológicos. El sistema permite registrar categorías, proveedores, productos, compras y ventas. Además, incluye consultas analíticas como productos más vendidos, stock por proveedor y categoría, productos con bajo stock y total de ventas mensuales por cliente. El sistema fue desarrollado con:

- Backend: Python + FastAPI + SQLAlchemy (Patrón Repository)
- Frontend: HTML + CSS + JavaScript
- Base de datos: MySQL Workbench

Una mejora aplicada fue que, al registrar una compra, el campo "cliente" se adapta para seleccionar entre los proveedores existentes, ya que en este contexto las compras representan adquisiciones hechas a proveedores registrados. Esto asegura la integridad y coherencia en los registros.

DER: Diagrama Entidad-Relación



Diccionario de Datos

Tabla	Campo	Tipo	Restricciones
Categoria	id	INT	PK, AUTO_INCREMENT
Categoria	nombre	VARCHAR(100)	NOT NULL, UNIQUE
Proveedor	id	INT	PK, AUTO_INCREMENT
Proveedor	nombre	VARCHAR(100)	NOT NULL
Proveedor	telefono	VARCHAR(20)	NULL
Proveedor	email	VARCHAR(100)	NULL
Producto	id	INT	PK, AUTO_INCREMENT
Producto	nombre	VARCHAR(100)	NOT NULL
Producto	descripcion	TEXT	NULL
Producto	precio	DECIMAL(10,2)	NOT NULL
Producto	stock	INT	NOT NULL, DEFAULT 0
Producto	categoria_id	INT	FK -> Categoria(id)
Producto	proveedor_id	INT	FK -> Proveedor(id)
Compra	id	INT	PK, AUTO_INCREMENT
Compra	producto_id	INT	FK -> Producto(id)
Compra	cantidad	INT	NOT NULL
Compra	cliente	VARCHAR(100)	NOT NULL (nombre del proveedor)
Compra	fecha	DATETIME	DEFAULT CURRENT_TIMESTAMP
Venta	id	INT	PK, AUTO_INCREMENT
Venta	producto_id	INT	FK -> Producto(id)
Venta	cantidad	INT	NOT NULL
Venta	cliente	VARCHAR(100)	NOT NULL

Venta	fecha	DATETIME	DEFAULT CURRENT_TIMESTAMP

Relaciones entre tablas

- Producto - Categoria: Cada producto pertenece a una categoría.
- Producto - Proveedor: Cada producto es suministrado por un proveedor.
- Producto - Compra: Cada compra se relaciona con un producto.
- Producto - Venta: Cada venta está asociada a un producto.
- Compra - Proveedor: La selección del cliente en compras corresponde a un proveedor registrado.

Estas relaciones permiten tener trazabilidad completa del stock y su movimiento.

Consultas Avanzadas

1. Productos más vendidos por categoría:
   Consulta que agrupa ventas por producto y categoría, sumando la cantidad vendida y ordenando por mayor venta.

```
SELECT

    c.nombre AS categoria,

    p.nombre AS producto,

    SUM(v.cantidad) AS total_vendido

FROM venta v

JOIN producto p ON v.producto_id = p.id

JOIN categoria c ON p.categoria_id = c.id

GROUP BY c.nombre, p.nombre

ORDER BY total_vendido DESC;
```

2. Stock actual por proveedor y categoría:
Consulta que suma el stock actual agrupado por proveedor y categoría de los productos.

```
SELECT
    pr.nombre AS proveedor,
    c.nombre AS categoria,
    SUM(p.stock) AS stock_total
FROM producto p
JOIN proveedor pr ON p.proveedor_id = pr.id
JOIN categoria c ON p.categoria_id = c.id
GROUP BY pr.nombre, c.nombre
ORDER BY pr.nombre, c.nombre;
```

3. Productos con menos de 5 unidades:
Consulta simple que lista los productos cuyo stock actual es menor que 5. Útil para

```
SELECT
    p.nombre AS producto,
    p.stock,
    c.nombre AS categoria,
    pr.nombre AS proveedor
FROM producto p
JOIN categoria c ON p.categoria_id = c.id
JOIN proveedor pr ON p.proveedor_id = pr.id
WHERE p.stock < 5
ORDER BY p.stock ASC;
```

4. Ventas mensuales por cliente:
Consulta que agrupa las ventas por cliente y mes (formateando la fecha), sumando las unidades compradas.

```
SELECT

    v.cliente,

    DATE_FORMAT(v.fecha, '%Y-%m') AS mes,

    SUM(v.cantidad) AS total_comprado

FROM venta v

GROUP BY v.cliente, mes

ORDER BY mes, v.cliente;
```

Estas consultas fueron implementadas mediante SQLAlchemy y devueltas como endpoints JSON al frontend, donde son mostradas en tablas dinámicas.