Instituto Politecnico Nacional
Escuela Superior de Computo

Bases de datos

Proyecto
Alumnos del equipo:

- Garcia Juarez hugo Armando
- Moreno Montes Aleexa Danae

Docente:

- Hurtado Aviles Gabriel

Repositorio:

https://github.com/danaemoremon/joyeria

Índice

Identificacion del problema *3*
Descripcion de la problematica 3
Requerimientos 4
Modelos entidad-relación *5*
Modelo entidad relacion 5
Modelo entidad relacion extendido 6
Modelo relacional 8
Codigo SQL *9*
Script 2 15
Pagina web *21*

Identificacion del problema

La tienda de joyería, que se dedica únicamente a vender piezas de plata, no cuenta con un sistema que le ayude a organizar bien su inventario, sus proveedores ni las ventas que realizan cada día. En lugar de usar un sistema digital, todo lo registran en hojas sueltas y cuadernos. Esto les ha traído varias complicaciones, porque muchas veces se pierde información importante o se repite, además de que al momento de contar las piezas se generan errores.

Por ejemplo, cuando llega un cliente y pregunta por un tipo específico de anillo o cadena, no siempre saben si realmente tienen la pieza disponible o cuántas quedan. También se vuelve difícil saber qué productos se están vendiendo más o cuáles ya se están terminando, porque tienen que revisar todo a mano y eso quita mucho tiempo. Con los proveedores pasa algo parecido: coo no hay un control bien organizado, se les complica dar seguimiento de quién surtió cada producto y en qué fecha.

En general, esta forma de trabajar les provoca retrasos, confusiones y hasta pérdidas económicas, porque no siempre pueden responder rápido a los clientes o reabastecer a tiempo lo que se acaba.

Descripcion de la problematica

Consultor: Buen día, gracias por aceptar esta entrevista. Me gustaría empezar preguntándole: ¿cuál es el problema principal que quisiera resolver si tuviera un sistema de base de datos?

***Cliente (dueño de la joyería*)**: Mire, lo que pasa es que yo trabajo únicamente con piezas de plata y tengo una gran variedad, como anillos, cadenas, aretes, pulseras y otros accesorios. El detalle es que llevo el control en hojas de cálculo y en cuadernos, pero eso ya me está causando problemas porque no siempre tengo la información actualizada. Muchas veces no sé cuántas piezas tengo realmente en existencia y cuando un cliente me pregunta por algo específico, tardo mucho en revisar mis apuntes o, de plano, no puedo darle una respuesta rápida. Eso hace que se pierdan ventas, que los clientes se desesperen y hasta se vayan a otra tienda.

Consultor: Entiendo. Y en cuanto a los productos, ¿qué tipo de información considera importante guardar en el sistema?

Cliente: Lo ideal sería poder registrar el nombre de cada pieza, el tipo al que pertenece (si es un anillo, una cadena, una pulsera, etc.), el costo de venta, la cantidad que tengo disponible en la tienda y además de qué proveedor viene cada una. Ahora lo apunto todo a mano y es fácil confundirse o que se me pase algo. Con un sistema sería más claro y rápido de consultar.

Consultor: Perfecto, eso suena muy necesario. Y dígame, ¿con qué frecuencia tendría que revisar esta información en el sistema?

Cliente: Todos los días, porque vendo constantemente y debo saber qué productos siguen en existencia. Por ejemplo, si alguien compra varias piezas en un mismo día, necesito que el inventario se actualice de inmediato para no ofrecer un producto que ya se acabó. También debo estar al tanto de qué debo pedir de nuevo a los proveedores para no quedarme sin stock y evitar que los clientes se queden sin lo que buscan.

Consultor: Comprendo. Por último, ¿qué tipo de reportes o consultas le gustaría poder obtener del sistema?

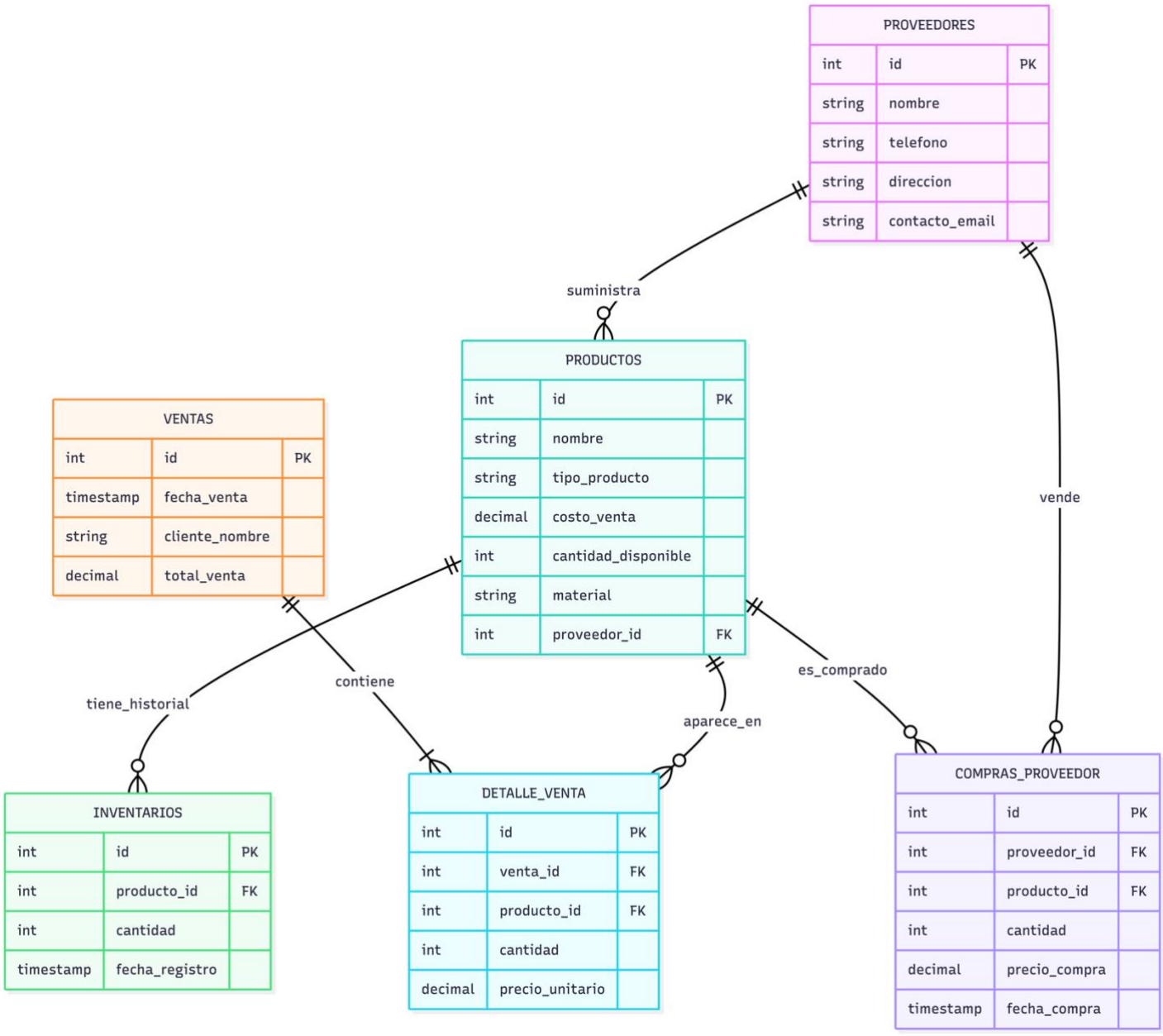
Cliente: Me interesa mucho saber qué productos se venden más, para darme cuenta de las tendencias y lo que más buscan mis clientes. También necesito que me avise cuáles joyas están por agotarse para reabastecer a tiempo. Otro punto importante es saber cuánto dinero ingreso en un periodo, como por semana o por mes, para llevar mejor mis cuentas y organizar mis ganancias. Y claro, me gustaría ver qué proveedores me surten más piezas y cuáles son los que más me conviene seguir utilizando a largo plazo.

Requerimientos

- El sistema debe guardar los datos de cada pieza de plata (nombre, tipo, costo de venta y cantidad disponible en la tienda).
- El sistema debe registrar de qué proveedor proviene cada pieza, así como su información básica (nombre y contacto).
- El sistema debe actualizar automáticamente el inventario cada vez que se realice una venta.
- Debe quedar guardado cada registro de venta (fecha, producto vendido, cantidad y total).
- El sistema debe permitir consultar las existencias de manera diaria para saber qué productos siguen disponibles.
- El sistema debe permitir registrar nuevas compras de inventario a los proveedores.
- El sistema debe permitir buscar productos de forma rápida (por nombre, tipo o proveedor).

Modelos entidad-relación

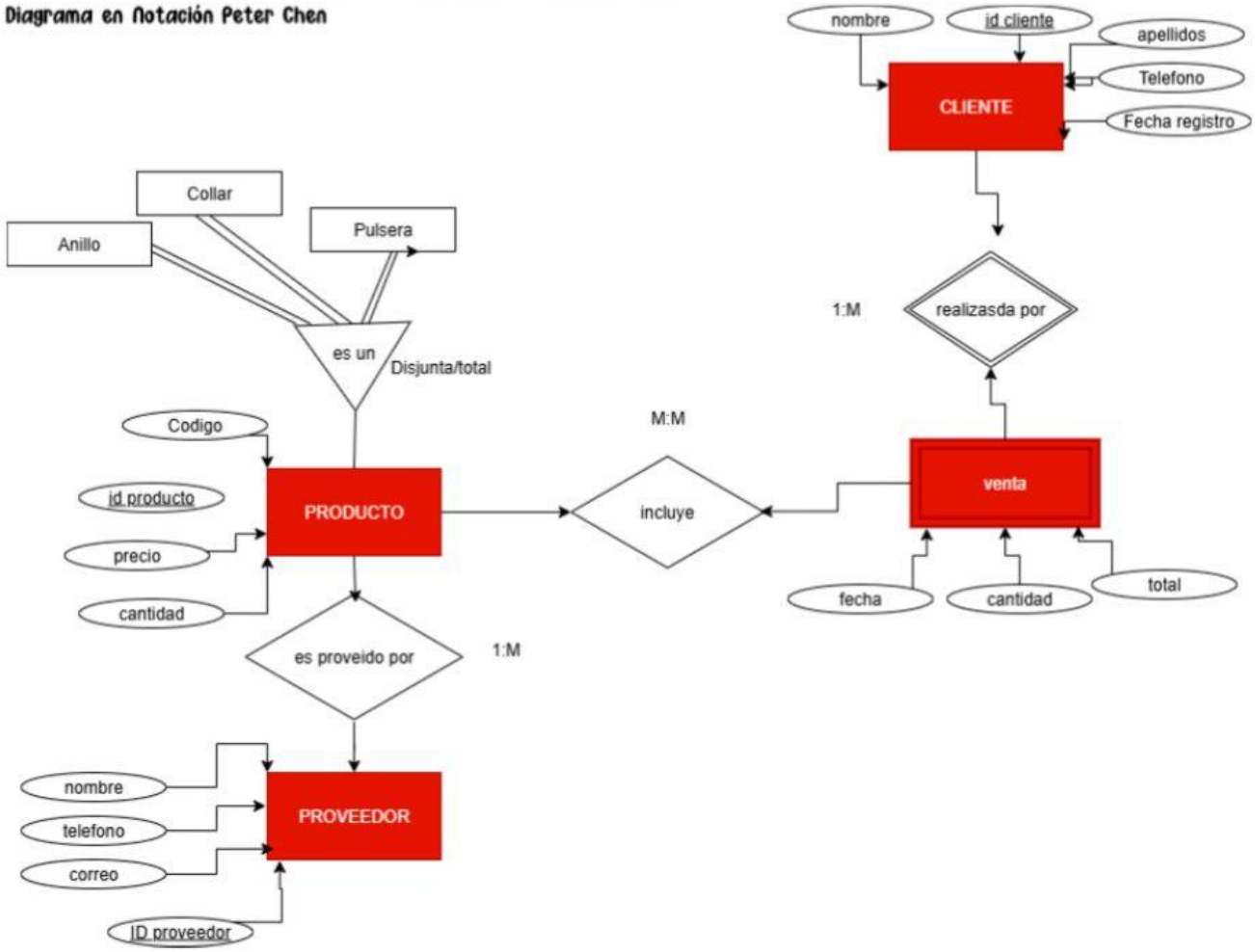
Modelo entidad relacion



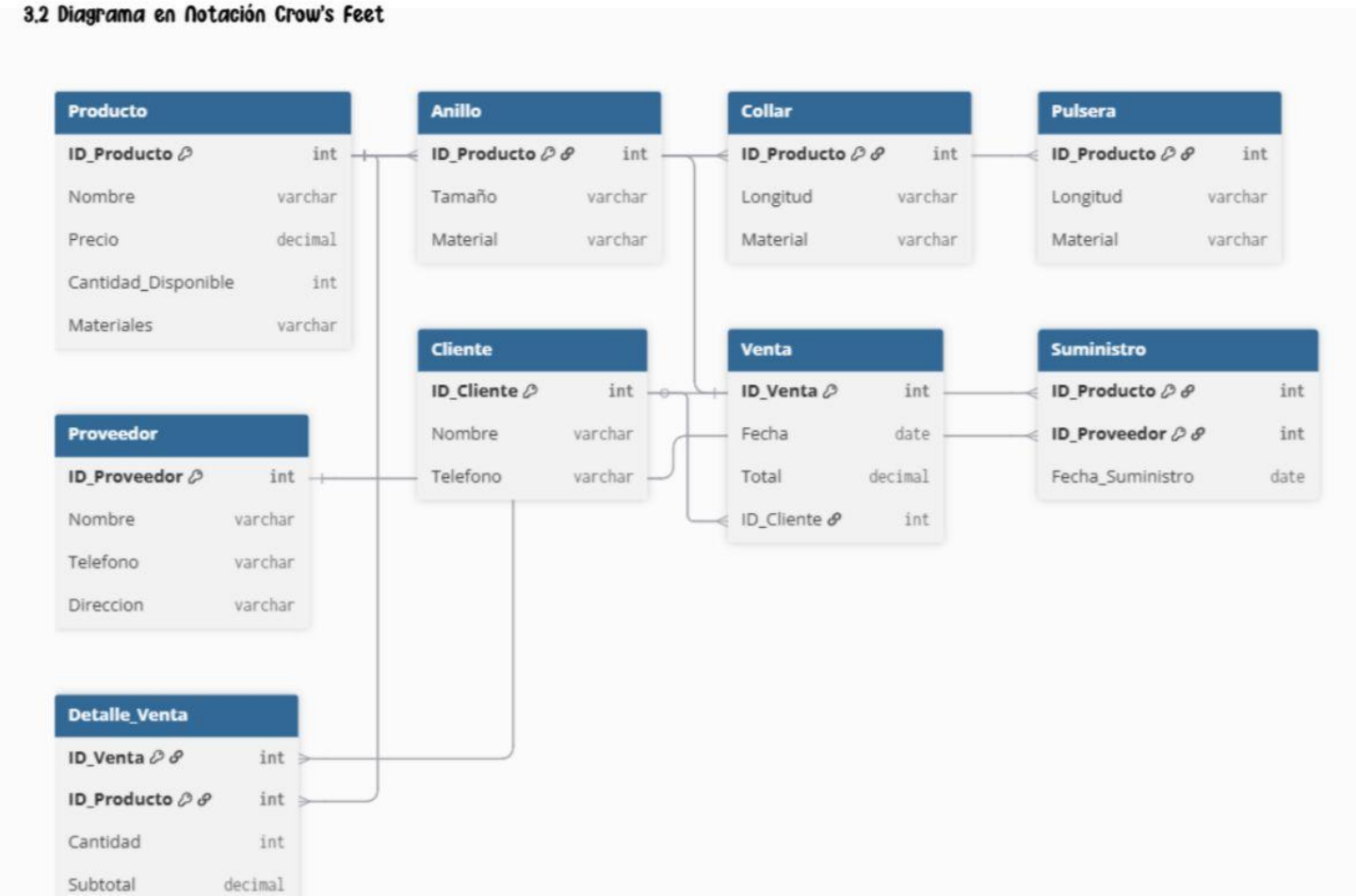
Modelo entidad relacion extendido

Ejercicio 3: Modelado EER con Múltiples Notaciones

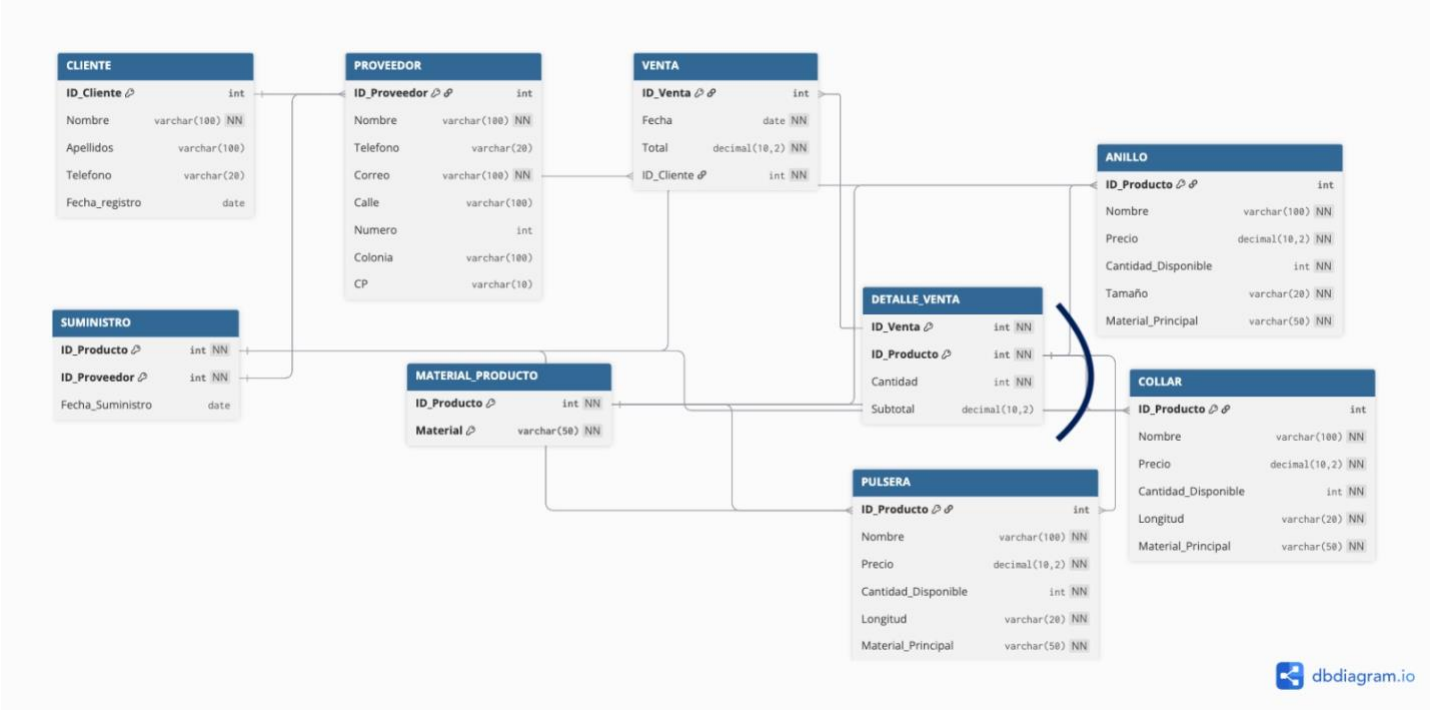
3.1 Diagrama en Notación Peter Chen



3.2 Diagrama en Notación Crow's Feet



Modelo relacional



Codigo SQL

```sql
-- phpMyAdmin SQL Dump
-- version 5.2.1
-- https://www.phpmyadmin.net/
--
-- Servidor: 127.0.0.1
-- Tiempo de generación: 07-01-2026 a las 01:58:19
-- Versión del servidor: 10.4.32-MariaDB
-- Versión de PHP: 8.2.12

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Base de datos: `u821594572_joyeria_db`
--

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `compras_proveedor`
--

CREATE TABLE `compras_proveedor` (
  `id` int(11) NOT NULL,
  `proveedor_id` int(11) DEFAULT NULL,
  `producto_id` int(11) DEFAULT NULL,
  `cantidad` int(11) NOT NULL,
  `precio_compra` decimal(10,2) NOT NULL,
  `fecha_compra` timestamp NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `detalle_venta`
```

```
--

CREATE TABLE `detalle_venta` (
  `id` int(11) NOT NULL,
  `venta_id` int(11) DEFAULT NULL,
  `producto_id` int(11) DEFAULT NULL,
  `cantidad` int(11) NOT NULL,
  `precio_unitario` decimal(10,2) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- --------------------------------------------------

--
-- Estructura de tabla para la tabla `inventarios`
--

CREATE TABLE `inventarios` (
  `id` int(11) NOT NULL,
  `producto_id` int(11) DEFAULT NULL,
  `cantidad` int(11) NOT NULL,
  `fecha_registro` timestamp NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- --------------------------------------------------

--
-- Estructura de tabla para la tabla `productos`
--

CREATE TABLE `productos` (
  `id` int(11) NOT NULL,
  `nombre` varchar(255) NOT NULL,
  `tipo_producto` varchar(100) NOT NULL,
  `costo_venta` decimal(10,2) NOT NULL,
  `cantidad_disponible` int(11) NOT NULL,
  `proveedor_id` int(11) DEFAULT NULL,
  `material` varchar(100) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

--
-- Volcado de datos para la tabla `productos`
--

INSERT INTO `productos` (`id`, `nombre`, `tipo_producto`, `costo_venta`,
`cantidad_disponible`, `proveedor_id`, `material`) VALUES
(1, 'Anillo Plata 925 Mujer', 'Anillo', 200.00, 12, 1, 'Plata 925'),
(2, 'Collar Cadena Plata 925', 'Collar', 100.00, 10, 1, 'Plata 925'),
```

```
(3, 'Pulsera Plata con Zirconias', 'Pulsera', 3000.00, 15, 1, 'Plata 925'),
(4, 'Anillo Corazón Plata 925', 'Anillo', 950.00, 8, 1, 'Plata 925'),
(5, 'Collar Doble Plata 925', 'Collar', 1150.00, 5, 1, 'Plata 925'),
(6, 'Pulsera Ajustable Plata 925', 'Pulsera', 680.00, 10, 1, 'Plata 925'),
(7, 'Anillo Cruz Plata 925', 'Anillo', 870.00, 7, 1, 'Plata 925'),
(8, 'Collar con Dije Luna Plata 925', 'Collar', 1100.00, 6, 1, 'Plata 925');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `proveedores`
--

CREATE TABLE `proveedores` (
  `id` int(11) NOT NULL,
  `nombre` varchar(255) NOT NULL,
  `telefono` varchar(15) DEFAULT NULL,
  `direccion` varchar(255) DEFAULT NULL,
  `contacto_email` varchar(255) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

--
-- Volcado de datos para la tabla `proveedores`
--

INSERT INTO `proveedores` (`id`, `nombre`, `telefono`, `direccion`,
`contacto_email`) VALUES
(1, 'Proveedor Plata MX', '5551234567', 'CDMX', 'ventas@plata.mx');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `usuarios`
--

CREATE TABLE `usuarios` (
  `id` int(11) NOT NULL,
  `nombre_usuario` varchar(50) NOT NULL,
  `email` varchar(100) NOT NULL,
  `contrasena` varchar(255) NOT NULL,
  `rol` enum('cliente','admin') DEFAULT 'cliente',
  `fecha_registro` timestamp NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

--
-- Volcado de datos para la tabla `usuarios`
--
```

```
INSERT INTO `usuarios` (`id`, `nombre_usuario`, `email`, `contrasena`, `rol`, `fecha_registro`) VALUES
(16, 'Hugolocal', 'Hugolocal@gmail.com', '$2y$10$zhaczH90hBfIIRtZfj5S7utN8Rs/DXKQise5ETwFp3zl6Ebu9gt3i', 'admin', '2025-11-14 20:16:31'),
(17, 'hugoad', 'hugoad@gmial.com', '$2y$10$RZVmDS0WXuJ.uO8wWuaLSOVOnHSLpk/FzZjVITeIMkpLCSesRGQUK', 'admin', '2025-12-19 15:46:22'),
(18, 'hugous', 'hugous@gmail.com', '$2y$10$DJj3wT4en8sRWFBwSKKqbedemjZHpmmvcFXVjylSbcGYHPvelm4Ju', 'cliente', '2025-12-19 16:14:43'),
(19, 'Dana', 'danapp@gmail.com', '$2y$10$alPJ5we8AktyYMIlqEuBre.d1/YDF4T4Zk25bBFS62E3ByyDHJ9vm', 'admin', '2026-01-07 00:47:48');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `ventas`
--

CREATE TABLE `ventas` (
  `id` int(11) NOT NULL,
  `fecha_venta` timestamp NULL DEFAULT current_timestamp(),
  `cliente_nombre` varchar(255) NOT NULL,
  `total_venta` decimal(10,2) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

--
-- Índices para tablas volcadas
--

--
-- Indices de la tabla `compras_proveedor`
--
ALTER TABLE `compras_proveedor`
  ADD PRIMARY KEY (`id`),
  ADD KEY `proveedor_id` (`proveedor_id`),
  ADD KEY `producto_id` (`producto_id`);

--
-- Indices de la tabla `detalle_venta`
--
ALTER TABLE `detalle_venta`
  ADD PRIMARY KEY (`id`),
  ADD KEY `venta_id` (`venta_id`),
```

```
  ADD KEY `producto_id` (`producto_id`);

--
-- Indices de la tabla `inventarios`
--
ALTER TABLE `inventarios`
  ADD PRIMARY KEY (`id`),
  ADD KEY `producto_id` (`producto_id`);

--
-- Indices de la tabla `productos`
--
ALTER TABLE `productos`
  ADD PRIMARY KEY (`id`),
  ADD KEY `proveedor_id` (`proveedor_id`)
--
-- Indices de la tabla `proveedores`
--
ALTER TABLE `proveedores`
  ADD PRIMARY KEY (`id`);

--
-- Indices de la tabla `usuarios`
--
ALTER TABLE `usuarios`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `nombre_usuario` (`nombre_usuario`),
  ADD UNIQUE KEY `email` (`email`);

--
-- Indices de la tabla `ventas`
--
ALTER TABLE `ventas`
  ADD PRIMARY KEY (`id`);

--
-- AUTO_INCREMENT de las tablas volcadas
--

--
-- AUTO_INCREMENT de la tabla `compras_proveedor`
--
ALTER TABLE `compras_proveedor`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT de la tabla `detalle_venta`
```

```sql
--
ALTER TABLE `detalle_venta`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT de la tabla `inventarios`
--
ALTER TABLE `inventarios`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT de la tabla `productos`
--
ALTER TABLE `productos`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=9;

--
-- AUTO_INCREMENT de la tabla `proveedores`
--
ALTER TABLE `proveedores`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;

--
-- AUTO_INCREMENT de la tabla `usuarios`
--
ALTER TABLE `usuarios`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=20;

--
-- AUTO_INCREMENT de la tabla `ventas`
--
ALTER TABLE `ventas`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;

--
-- Restricciones para tablas volcadas
--


--
-- Filtros para la tabla `compras_proveedor`
--
ALTER TABLE `compras_proveedor`
  ADD CONSTRAINT `compras_proveedor_ibfk_1` FOREIGN KEY (`proveedor_id`)
REFERENCES `proveedores` (`id`) ON DELETE CASCADE,
  ADD CONSTRAINT `compras_proveedor_ibfk_2` FOREIGN KEY (`producto_id`)
REFERENCES `productos` (`id`) ON DELETE CASCADE;
```

```
--
-- Filtros para la tabla `detalle_venta`
--
ALTER TABLE `detalle_venta`
  ADD CONSTRAINT `detalle_venta_ibfk_1` FOREIGN KEY (`venta_id`) REFERENCES `ventas` (`id`) ON DELETE CASCADE,
  ADD CONSTRAINT `detalle_venta_ibfk_2` FOREIGN KEY (`producto_id`) REFERENCES `productos` (`id`) ON DELETE CASCADE;

--
-- Filtros para la tabla `inventarios`
--
ALTER TABLE `inventarios`
  ADD CONSTRAINT `inventarios_ibfk_1` FOREIGN KEY (`producto_id`) REFERENCES `productos` (`id`) ON DELETE CASCADE;

--
-- Filtros para la tabla `productos`
--
ALTER TABLE `productos`
  ADD CONSTRAINT `productos_ibfk_1` FOREIGN KEY (`proveedor_id`) REFERENCES `proveedores` (`id`) ON DELETE SET NULL;
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

Script 2

Está diseñado para ser re-ejecutable: utiliza DROP TABLE...CASCADE para eliminar las tablas en el orden correcto, sin importar las dependencias, antes de volver a crearlas.

Implementa todas las restricciones de los Ejercicios 2 y 3.

```
-- ELIMINACIÓN SEGURA DE OBJETOS (Permite re-ejecución)
-- Usamos CASCADE para eliminar las dependencias (FKs) antes de borrar la tabla.
DROP TABLE IF EXISTS DETALLE_VENTA;
DROP TABLE IF EXISTS COMPRAS_PROVEEDOR;
```

```sql
DROP TABLE IF EXISTS INVENTARIOS;
DROP TABLE IF EXISTS VENTAS;
DROP TABLE IF EXISTS ANILLO;
DROP TABLE IF EXISTS COLLAR;
DROP TABLE IF EXISTS PULSERA;
DROP TABLE IF EXISTS PRODUCTOS;
DROP TABLE IF EXISTS PROVEEDORES;
DROP TABLE IF EXISTS CLIENTE;
-----------------------------------------------------------------
-- SECCIÓN 1: Creación de Entidades Padre y Superclases
-----------------------------------------------------------------
-- 1. CLIENTE (Entidad Padre)
CREATE TABLE CLIENTE (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    apellidos VARCHAR(100),
    telefono VARCHAR(20),
    email VARCHAR(100) UNIQUE -- Restricción UNIQUE (2.4)
);

-- 2. PROVEEDORES (Entidad Padre)
CREATE TABLE PROVEEDORES (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(255) NOT NULL,
    telefono VARCHAR(15),
    direccion VARCHAR(255),
    contacto_email VARCHAR(255) UNIQUE -- Restricción UNIQUE (2.4)
);

-- 3. PRODUCTOS (Superclase, Hija de PROVEEDORES)
CREATE TABLE PRODUCTOS (
    id SERIAL PRIMARY KEY,
    proveedor_id INT NOT NULL,
    nombre VARCHAR(255) NOT NULL,
    costo_venta DECIMAL(10, 2) NOT NULL,

    -- Discriminador para la especialización
    tipo_producto VARCHAR(50) NOT NULL,

    -- Restricciones CHECK (2.5)
    CONSTRAINT chk_producto_costo CHECK (costo_venta > 0),
    CONSTRAINT chk_producto_tipo CHECK (tipo_producto IN ('Anillo', 'Collar',
'Pulsera')),

    -- Clave Foránea (3.1 y 3.2)
    CONSTRAINT fk_producto_proveedor
        FOREIGN KEY (proveedor_id) REFERENCES PROVEEDORES(id)
```

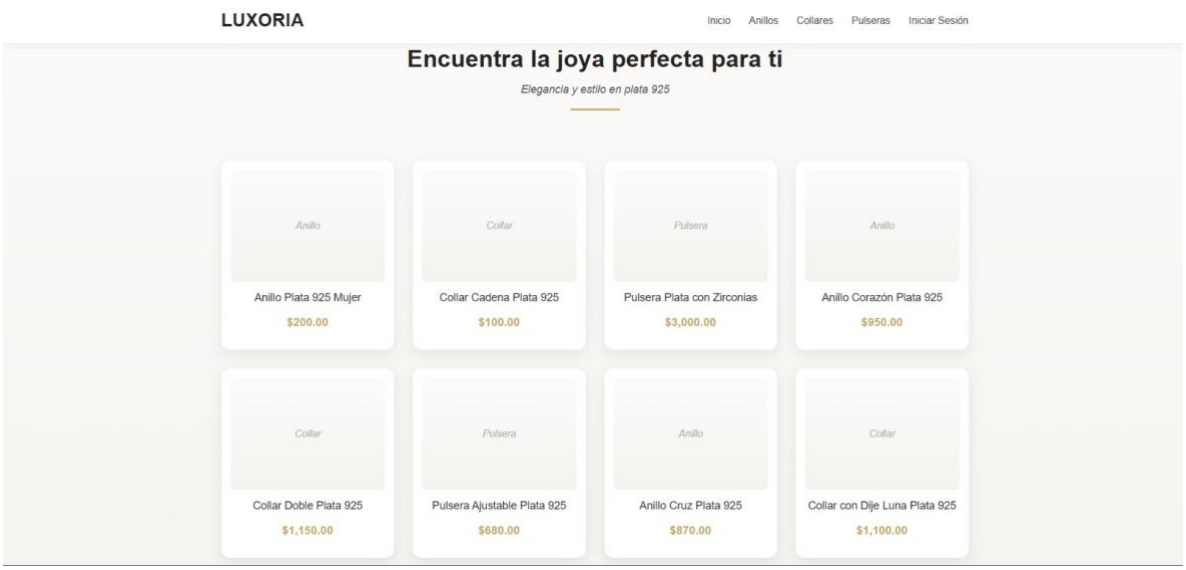
```
        ON DELETE RESTRICT -- No borrar proveedor si tiene productos
        ON UPDATE CASCADE
);


-----------------------------------------------------------------
-- SECCIÓN 2: Creación de Subclases (Especialización de PRODUCTOS)
-- Estrategia A: Supertipo/Subtipo
-----------------------------------------------------------------

-- 4. ANILLO (Subclase de PRODUCTOS)
CREATE TABLE ANILLO (
    producto_id INT PRIMARY KEY, -- Es PK y FK
    material VARCHAR(100),
    tamaño DECIMAL(4, 1),

    -- Restricción CHECK (2.5)
    CONSTRAINT chk_anillo_tamaño CHECK (tamaño > 0),

    -- Clave Foránea (3.1 y 3.2) - Dependencia Fuerte (Herencia)
    CONSTRAINT fk_anillo_producto
        FOREIGN KEY (producto_id) REFERENCES PRODUCTOS(id)
        ON DELETE CASCADE -- Si el PRODUCTO se borra, el ANILLO (que es
parte) se borra
        ON UPDATE CASCADE
);

-- 5. COLLAR (Subclase de PRODUCTOS)
CREATE TABLE COLLAR (
    producto_id INT PRIMARY KEY, -- Es PK y FK
    material VARCHAR(100),
    longitud_cm INT,

    -- Restricción CHECK (2.5)
    CONSTRAINT chk_collar_longitud CHECK (longitud_cm > 0),

    -- Clave Foránea (3.1 y 3.2) - Dependencia Fuerte (Herencia)
    CONSTRAINT fk_collar_producto
        FOREIGN KEY (producto_id) REFERENCES PRODUCTOS(id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

-- 6. PULSERA (Subclase de PRODUCTOS)
CREATE TABLE PULSERA (
    producto_id INT PRIMARY KEY, -- Es PK y FK
    material VARCHAR(100),
```

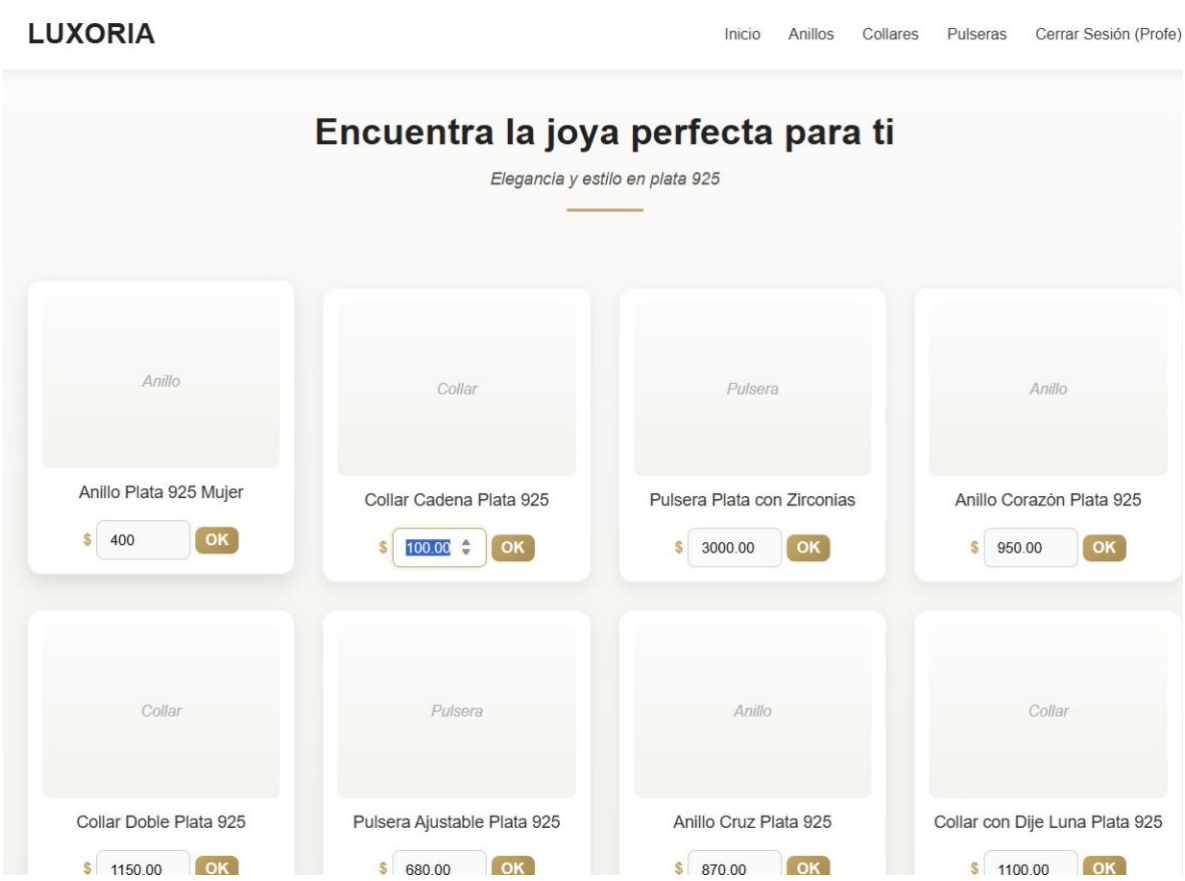
```
    longitud_cm INT,

    -- Restricción CHECK (2.5)
    CONSTRAINT chk_pulsera_longitud CHECK (longitud_cm > 0),

    -- Clave Foránea (3.1 y 3.2) - Dependencia Fuerte (Herencia)
    CONSTRAINT fk_pulsera_producto
        FOREIGN KEY (producto_id) REFERENCES PRODUCTOS(id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

-- ----------------------------------------------------------------
-- SECCIÓN 3: Creación de Entidades Transaccionales y Asociativas
-- ----------------------------------------------------------------

-- 7. VENTAS (Hija de CLIENTE)
CREATE TABLE VENTAS (
    id SERIAL PRIMARY KEY,
    cliente_id INT NOT NULL,
    fecha_venta TIMESTAMP DEFAULT CURRENT_TIMESTAMP, -- Restricción
DEFAULT (2.3)
    total_venta DECIMAL(10, 2) NOT NULL,

    -- Restricción CHECK (2.5)
    CONSTRAINT chk_venta_total CHECK (total_venta >= 0),

    -- Clave Foránea (3.1 y 3.2)
    CONSTRAINT fk_venta_cliente
        FOREIGN KEY (cliente_id) REFERENCES CLIENTE(id)
        ON DELETE RESTRICT -- No borrar cliente si tiene ventas (historial)
        ON UPDATE CASCADE
);

-- 8. INVENTARIOS (Hijo de PRODUCTOS)
CREATE TABLE INVENTARIOS (
    id SERIAL PRIMARY KEY,
    producto_id INT NOT NULL UNIQUE, -- Restricción UNIQUE (2.4)
    cantidad INT NOT NULL DEFAULT 0, -- Restricción DEFAULT (2.3)
    fecha_registro TIMESTAMP DEFAULT CURRENT_TIMESTAMP, -- Restricción
DEFAULT (2.3)

    -- Restricción CHECK (2.5)
    CONSTRAINT chk_inventario_cantidad CHECK (cantidad >= 0),

    -- Clave Foránea (3.1 y 3.2) - Dependencia Fuerte (Composición)
    CONSTRAINT fk_inventario_producto
```

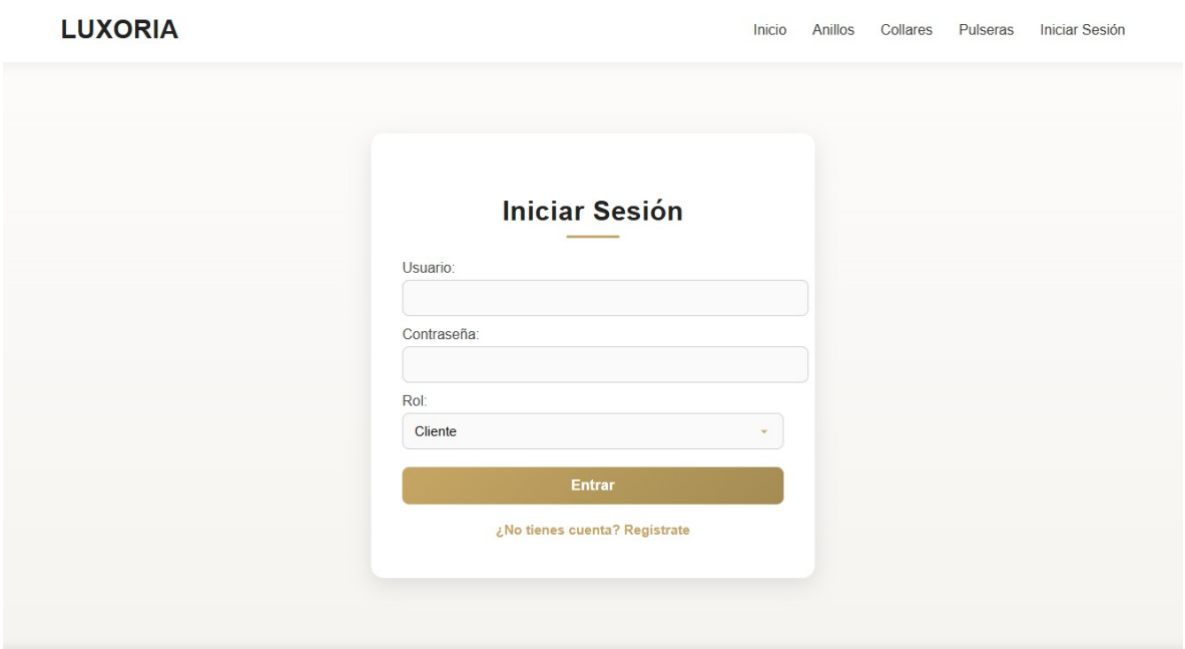
```
        FOREIGN KEY (producto_id) REFERENCES PRODUCTOS(id)
        ON DELETE CASCADE -- Si el producto se borra, su registro de inventario también
        ON UPDATE CASCADE
);

-- 9. COMPRAS_PROVEEDOR (Tabla Asociativa N:M)
CREATE TABLE COMPRAS_PROVEEDOR (
    id SERIAL PRIMARY KEY,
    proveedor_id INT NOT NULL,
    producto_id INT NOT NULL,
    cantidad INT NOT NULL,
    precio_compra DECIMAL(10, 2) NOT NULL,
    fecha_compra TIMESTAMP DEFAULT CURRENT_TIMESTAMP, -- Restricción DEFAULT (2.3)

    -- Restricciones CHECK (2.5)
    CONSTRAINT chk_compra_cantidad CHECK (cantidad > 0),
    CONSTRAINT chk_compra_precio CHECK (precio_compra > 0),

    -- Claves Foráneas (3.1 y 3.2)
    CONSTRAINT fk_compra_proveedor
        FOREIGN KEY (proveedor_id) REFERENCES PROVEEDORES(id)
        ON DELETE RESTRICT -- Proteger historial de compras
        ON UPDATE CASCADE,

    CONSTRAINT fk_compra_producto
        FOREIGN KEY (producto_id) REFERENCES PRODUCTOS(id)
        ON DELETE RESTRICT -- Proteger historial de compras
        ON UPDATE CASCADE
);

-- 10. DETALLE_VENTA (Tabla Asociativa N:M)
CREATE TABLE DETALLE_VENTA (
    id SERIAL PRIMARY KEY,
    venta_id INT NOT NULL,
    producto_id INT NOT NULL,
    cantidad INT NOT NULL,
    precio_unitario DECIMAL(10, 2) NOT NULL,

    -- Restricción UNIQUE Compuesta (2.4)
    CONSTRAINT uq_venta_producto UNIQUE (venta_id, producto_id),

    -- Restricciones CHECK (2.5)
    CONSTRAINT chk_detalle_cantidad CHECK (cantidad > 0),
    CONSTRAINT chk_detalle_precio CHECK (precio_unitario >= 0),
```

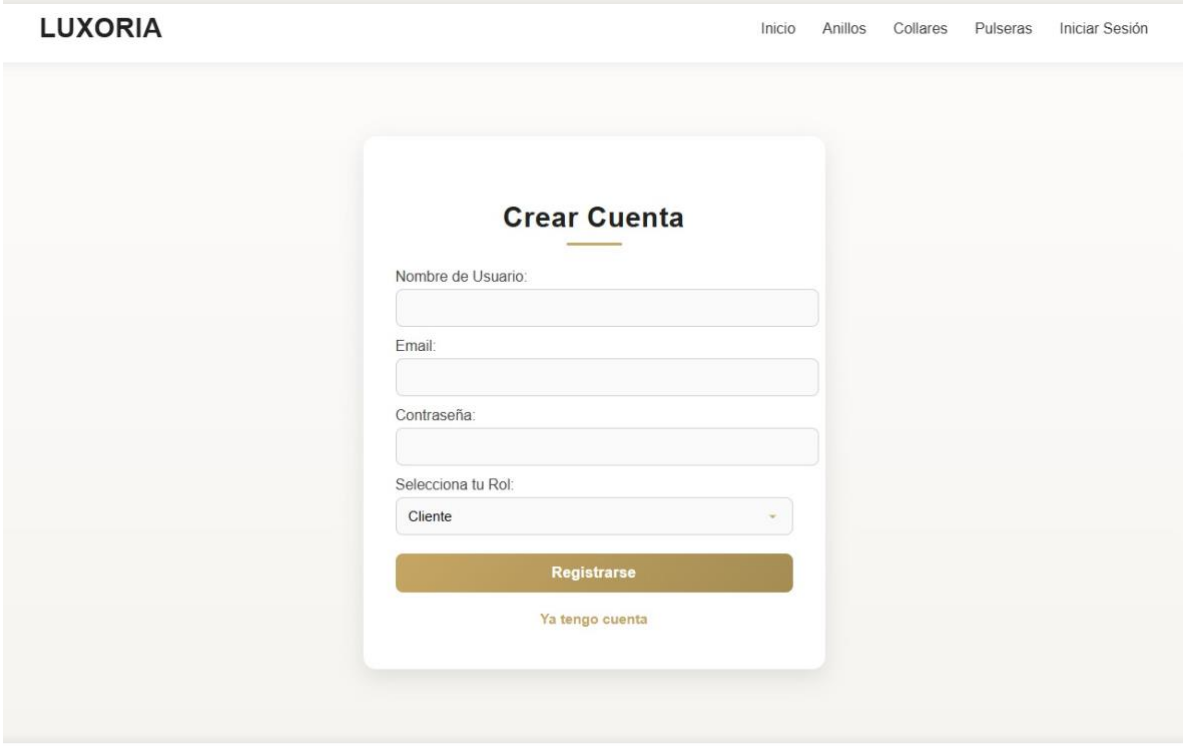
```
    -- Claves Foráneas (3.1 y 3.2)
    CONSTRAINT fk_detalle_venta
        FOREIGN KEY (venta_id) REFERENCES VENTAS(id)
        ON DELETE CASCADE -- Si la Venta se anula, el detalle se borra
        ON UPDATE CASCADE,

    CONSTRAINT fk_detalle_producto
        FOREIGN KEY (producto_id) REFERENCES PRODUCTOS(id)
        ON DELETE RESTRICT -- No borrar producto si tiene historial de ventas
        ON UPDATE CASCADE
);
```

Pagina web


LUXORIA
Inicio
Anillos
Collares
Pulseras
Iniciar Sesión
Encuentra la joya perfecta para ti
Elegancia y estilo en plata 925
Anillo
Anillo Plata 925 Mujer
$200.00
Collar
Collar Cadena Plata 925
$100.00
Pulsera
Pulsera Plata con Zirconias
$3,000.00
Anillo
Anillo Corazón Plata 925
$950.00
Collar
Collar Doble Plata 925
$1,150.00
Pulsera
Pulsera Ajustable Plata 925
$680.00
Anillo
Anillo Cruz Plata 925
$870.00
Collar
Collar con Dije Luna Plata 925
$1,100.00


LUXORIA
Inicio
Anillos
Collares
Pulseras
Cerrar Sesión (Profe)
Encuentra la joya perfecta para ti
Elegancia y estilo en plata 925
Anillo
Anillo Plata 925 Mujer
$ 400 OK
Collar
Collar Cadena Plata 925
$ 100.00 OK
Pulsera
Pulsera Plata con Zirconias
$ 3000.00 OK
Anillo
Anillo Corazón Plata 925
$ 950.00 OK
Collar
Collar Doble Plata 925
$ 1150.00 OK
Pulsera
Pulsera Ajustable Plata 925
$ 680.00 OK
Anillo
Anillo Cruz Plata 925
$ 870.00 OK
Collar
Collar con Dije Luna Plata 925
$ 1100.00 OK


LUXORIA
Inicio
Anillos
Collares
Pulseras
Iniciar Sesión
Iniciar Sesión
Usuario:
Contraseña:
Rol:
Cliente
Entrar
¿No tienes cuenta? Regístrate


LUXORIA
Inicio
Anillos
Collares
Pulseras
Iniciar Sesión
Crear Cuenta
Nombre de Usuario:
Email:
Contraseña:
Selecciona tu Rol:
Cliente
Registrarse
Ya tengo cuenta

Conclusiones

El desarrollo del proyecto Luxoria ha permitido la creación de una solución tecnológica robusta y funcional para la administración de una tienda de joyería fina, logrando integrar con éxito la gestión de inventarios, usuarios y transacciones comerciales. A través de este trabajo, se han alcanzado los siguientes hitos fundamentales:

1. **Eficiencia en la Gestión de Datos:** Mediante la implementación de una base de datos relacional normalizada en MariaDB/MySQL, el sistema garantiza la integridad de la información. La estructura de tablas permite un control preciso sobre el stock de productos de plata 925, la relación con proveedores y el historial detallado de ventas.
2. **Seguridad y Roles de Acceso:** Se logró establecer un entorno seguro mediante el uso de algoritmos de encriptación BCRYPT para contraseñas y sentencias preparadas para mitigar vulnerabilidades como la inyección SQL. Además, la diferenciación de roles (Administrador y Cliente) asegura que solo el personal autorizado pueda modificar precios y existencias, mientras que el cliente disfruta de una interfaz de consulta fluida.
3. **Arquitectura Moderna:** El uso de PHP 8.x junto con Doctrine ORM ha permitido transicionar de un esquema de programación estructurada a uno orientado a objetos más limpio y mantenible. La integración de Composer para la gestión de dependencias y Symfony Cache para el rendimiento posiciona al proyecto bajo estándares profesionales de desarrollo web.
4. **Experiencia de Usuario (UX/UI):** El frontend, desarrollado con HTML5, CSS3 y la tipografía Poppins, ofrece una interfaz elegante y minimalista acorde al mercado de lujo de la joyería. La capacidad de actualización de precios en tiempo real desde el dashboard administrativo optimiza significativamente los tiempos de operación del negocio.

En conclusión, Luxoria representa una herramienta integral que no solo automatiza procesos administrativos complejos, sino que también ofrece una base escalable para futuras implementaciones, como módulos de pagos en línea o facturación electrónica, cumpliendo satisfactoriamente con los objetivos planteados para el curso de Bases de Datos 2026-I.