**DESCRIPCIÓN GENERAL DEL PROYECTO**

Aplicación integral para la gestión de inventario, ventas y servicio post-venta de prótesis médicas. El sistema debe permitir el control eficiente del inventario especializado, gestionar el proceso de ventas considerando las particularidades del producto, y ofrecer un seguimiento adecuado del servicio post-venta, incluyendo devoluciones y cambios que cumplan con las normativas sanitarias correspondientes.

**MÓDULOS REQUERIDOS**

**1. MÓDULO DE INVENTARIO**

**Funcionalidades Principales**

* **Catalogación de Prótesis**
  + Clasificación por tipo (ortopédicas, dentales, oculares, etc.)
  + Registro de especificaciones técnicas y materiales
  + Control de números de serie, lotes y fechas de fabricación
  + Gestión de certificaciones y documentación técnica asociada
  + Registro de proveedores por tipo de prótesis
* **Control de Stock**
  + Seguimiento en tiempo real de existencias
  + Alertas de stock mínimo configurable por categoría
  + Gestión de ubicaciones físicas (almacenes, estanterías, etc.)
  + Trazabilidad completa desde recepción hasta entrega
* **Gestión de Caducidad y Esterilización**
  + Control de fechas de vencimiento y alertas automáticas
  + Registro de procesos de esterilización cuando aplique
  + Etiquetado con códigos QR/barras para seguimiento
* **Recepción de Mercancía**
  + Validación contra órdenes de compra
  + Inspección de calidad y registro fotográfico
  + Cuarentena para productos que requieren validación adicional

**Mejores Prácticas**

* Implementar sistema FEFO (First Expired, First Out) para gestión de stock
* Integración con lectores de códigos de barras/QR para minimizar errores
* Implementación de historiales de auditoría para cada movimiento de inventario
* Reportes automáticos de rotación de inventario y proyecciones de demanda
* Gestión de productos consignados con seguimiento especial

**2. MÓDULO DE VENTAS**

**Funcionalidades Principales**

* **Gestión de Clientes**
  + Registro detallado de información de pacientes y médicos
  + Historial de adquisiciones y prescripciones médicas
  + Gestión de seguros médicos y coberturas
  + Seguimiento de preferencias y necesidades específicas
* **Proceso de Venta**
  + Generación de cotizaciones con validez temporal
  + Configurador de prótesis personalizadas según especificaciones médicas
  + Registro de prescripciones médicas y documentación requerida
  + Cálculo automático de precios según configuraciones y personalizaciones
  + Gestión de anticipos y pagos parciales
* **Facturación y Pagos**
  + Integración con sistemas fiscales locales
  + Gestión de múltiples formas de pago
  + Manejo de descuentos autorizados y convenios institucionales
  + Facturación a aseguradoras y entidades de salud
* **Agenda y Seguimiento**
  + Programación de citas para toma de medidas
  + Seguimiento de fabricación para prótesis personalizadas
  + Calendarización de entregas y pruebas de ajuste

**Mejores Prácticas**

* Implementación de firma electrónica para aprobaciones y consentimientos
* Dashboards de ventas con KPIs relevantes para el sector
* Integración con CRM para seguimiento de oportunidades de venta
* Gestión de comisiones para vendedores y referenciadores médicos
* Trazabilidad completa de interacciones con clientes (enfoque médico)

**3. MÓDULO DE POST-VENTA**

**Funcionalidades Principales**

* **Seguimiento de Adaptación**
  + Registro de seguimientos post-implantación/colocación
  + Alertas para controles programados según tipo de prótesis
  + Documentación de evolución y adaptación del paciente
  + Gestión de ajustes menores incluidos en garantía
* **Servicio Técnico**
  + Registro de incidencias y solicitudes técnicas
  + Seguimiento de reparaciones y mantenimientos
  + Control de piezas de repuesto utilizadas
  + Gestión de tiempos de respuesta según prioridades médicas
* **Garantías**
  + Registro de condiciones específicas por tipo de producto
  + Trazabilidad de plazos y coberturas
  + Documentación de procedimientos realizados bajo garantía
  + Alertas de vencimiento de garantías

**Mejores Prácticas**

* Encuestas de satisfacción específicas para el sector protésico
* Sistema de tickets para seguimiento de casos post-venta
* Biblioteca de conocimiento para resolución de problemas comunes
* Calendario de seguimiento automático según protocolos médicos
* Integración con plataformas de telemedicina para seguimientos remotos

**4. MÓDULO DE DEVOLUCIONES Y CAMBIOS**

**Funcionalidades Principales**

* **Gestión de Devoluciones**
  + Registro de motivos médicos o técnicos
  + Evaluación de estado del producto según normativas sanitarias
  + Trazabilidad de autorizaciones médicas para devoluciones
  + Gestión de reembolsos y notas de crédito
* **Proceso de Cambios**
  + Validación de compatibilidad para cambios de modelo/tamaño
  + Registro de ajustes realizados en productos devueltos
  + Control de diferencias de precio en cambios
  + Documentación técnica de modificaciones realizadas
* **Disposición Final**
  + Protocolos de destrucción para productos no reutilizables
  + Registro de procesos de reacondicionamiento cuando aplique
  + Certificación de eliminación segura según regulaciones
  + Trazabilidad completa para auditorías sanitarias

**Mejores Prácticas**

* Implementación de flujos de aprobación con múltiples niveles para casos especiales
* Registro fotográfico de estado de productos devueltos
* Integración con normativas sanitarias locales sobre reutilización de dispositivos médicos
* Reportes estadísticos de causas de devolución para mejora continua
* Gestión documental de certificados de destrucción/reacondicionamiento

**5. MÓDULO DE REPORTES Y ANALÍTICA**

**Funcionalidades Principales**

* **Reportes Operativos**
  + Rotación de inventario por categorías
  + Estadísticas de ventas por tipo de prótesis, región y especialista
  + Análisis de devoluciones y sus causas
  + Tiempos de entrega y cumplimiento de plazos
* **Reportes Financieros**
  + Rentabilidad por línea de producto
  + Análisis de costos de garantías y servicio post-venta
  + Proyecciones de ingresos basadas en históricos
  + Control de márgenes por producto y cliente
* **Analítica Avanzada**
  + Predicción de demanda por temporalidad y factores externos
  + Segmentación de clientes por valor y frecuencia
  + Análisis de satisfacción y correlación con productos específicos
  + Identificación de productos con mayor incidencia de problemas

**Mejores Prácticas**

* Dashboards interactivos con filtros contextuales
* Exportación a múltiples formatos (PDF, Excel, CSV)
* Programación de envío automático de reportes clave
* Visualizaciones específicas para comités médicos y técnicos
* Análisis comparativos con benchmarks del sector

**REQUISITOS TÉCNICOS**

**Arquitectura del Sistema**

* **Frontend**: Aplicación web responsive con enfoque mobile-first
* **Backend**: API RESTful con autenticación OAuth 2.0 y JWT
* **Base de Datos**: MySQL con replicación y alta disponibilidad
* **Integraciones**:
  + Pasarelas de pago
  + Sistemas fiscales
  + APIs de aseguradoras (cuando estén disponibles)
  + Sistemas de logística para seguimiento de envíos

**Seguridad**

* Cumplimiento con regulaciones de datos personales y médicos (GDPR, HIPAA, etc.)
* Encriptación de datos sensibles en reposo y en tránsito
* Autenticación de doble factor para accesos críticos
* Registro de auditoría detallado para cambios en datos sensibles
* Perfiles de usuario con privilegios granulares

**Experiencia de Usuario**

* Interfaz intuitiva adaptada a personal médico y técnico
* Visualización de modelos 3D para prótesis cuando sea posible
* Flujos de trabajo optimizados para reducir clics en operaciones frecuentes
* Sistema de ayuda contextual y tutoriales integrados
* Compatibilidad con dispositivos médicos especializados (escáneres 3D, etc.)

**DISEÑO DE BASE DE DATOS MYSQL**

A continuación se presenta el esquema relacional para la base de datos del sistema, organizado por módulos:

**Esquema de Tablas - Inventario**

-- Categorías de prótesis

CREATE TABLE categories (

category\_id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

description TEXT,

parent\_category\_id INT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (parent\_category\_id) REFERENCES categories(category\_id)

);

**-- Proveedores**

CREATE TABLE suppliers (

supplier\_id INT AUTO\_INCREMENT PRIMARY KEY,

company\_name VARCHAR(100) NOT NULL,

contact\_name VARCHAR(100),

email VARCHAR(100),

phone VARCHAR(20),

address TEXT,

tax\_id VARCHAR(50),

status ENUM('active', 'inactive') DEFAULT 'active',

notes TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

**-- Productos (Prótesis)**

CREATE TABLE products (

product\_id INT AUTO\_INCREMENT PRIMARY KEY,

sku VARCHAR(50) UNIQUE NOT NULL,

name VARCHAR(150) NOT NULL,

category\_id INT NOT NULL,

description TEXT,

technical\_specs TEXT,

materials TEXT,

requires\_prescription BOOLEAN DEFAULT TRUE,

is\_customizable BOOLEAN DEFAULT FALSE,

production\_time\_days INT DEFAULT 0,

min\_stock INT DEFAULT 5,

certification\_info TEXT,

weight\_grams DECIMAL(10,2),

dimensions VARCHAR(50),

image\_url VARCHAR(255),

status ENUM('active', 'discontinued', 'pending\_approval') DEFAULT 'active',

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (category\_id) REFERENCES categories(category\_id)

);

**-- Variantes de producto (tallas, configuraciones)**

CREATE TABLE product\_variants (

variant\_id INT AUTO\_INCREMENT PRIMARY KEY,

product\_id INT NOT NULL,

size VARCHAR(50),

color VARCHAR(50),

material\_variant VARCHAR(100),

additional\_specs TEXT,

sku\_variant VARCHAR(60) UNIQUE NOT NULL,

price\_adjustment DECIMAL(10,2) DEFAULT 0.00,

status ENUM('active', 'discontinued') DEFAULT 'active',

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (product\_id) REFERENCES products(product\_id)

);

**-- Inventario (stock por ubicación)**

CREATE TABLE inventory\_items (

inventory\_id INT AUTO\_INCREMENT PRIMARY KEY,

variant\_id INT NOT NULL,

location\_id INT NOT NULL,

batch\_number VARCHAR(50),

serial\_number VARCHAR(100),

quantity INT NOT NULL DEFAULT 0,

manufacturing\_date DATE,

expiration\_date DATE,

sterilization\_date DATE,

last\_quality\_check DATE,

purchase\_order\_id INT,

unit\_cost DECIMAL(10,2),

status ENUM('available', 'reserved', 'quarantine', 'defective') DEFAULT 'available',

notes TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (variant\_id) REFERENCES product\_variants(variant\_id),

FOREIGN KEY (location\_id) REFERENCES locations(location\_id)

);

**-- Ubicaciones físicas**

CREATE TABLE locations (

location\_id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

type ENUM('warehouse', 'store', 'transit') NOT NULL,

address TEXT,

contact\_person VARCHAR(100),

phone VARCHAR(20),

is\_active BOOLEAN DEFAULT TRUE,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

**-- Movimientos de inventario**

CREATE TABLE inventory\_movements (

movement\_id INT AUTO\_INCREMENT PRIMARY KEY,

inventory\_id INT NOT NULL,

type ENUM('purchase', 'sale', 'transfer', 'return', 'adjustment', 'disposal') NOT NULL,

quantity INT NOT NULL,

from\_location\_id INT,

to\_location\_id INT,

reference\_id INT COMMENT 'ID of related document (order, transfer, etc)',

reference\_type VARCHAR(50) COMMENT 'Type of reference (order, transfer, etc)',

user\_id INT NOT NULL,

movement\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

reason TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (inventory\_id) REFERENCES inventory\_items(inventory\_id),

FOREIGN KEY (from\_location\_id) REFERENCES locations(location\_id),

FOREIGN KEY (to\_location\_id) REFERENCES locations(location\_id)

);

**Esquema de Tablas - Ventas**

-- Clientes (pacientes, médicos, instituciones)

CREATE TABLE customers (

customer\_id INT AUTO\_INCREMENT PRIMARY KEY,

customer\_type ENUM('patient', 'doctor', 'institution') NOT NULL,

first\_name VARCHAR(100),

last\_name VARCHAR(100),

company\_name VARCHAR(150),

email VARCHAR(100),

phone VARCHAR(20),

address TEXT,

tax\_id VARCHAR(50),

date\_of\_birth DATE,

medical\_record\_id VARCHAR(100),

insurance\_info TEXT,

credit\_limit DECIMAL(10,2) DEFAULT 0,

status ENUM('active', 'inactive') DEFAULT 'active',

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

**-- Relaciones médico-paciente**

CREATE TABLE doctor\_patient (

id INT AUTO\_INCREMENT PRIMARY KEY,

doctor\_id INT NOT NULL,

patient\_id INT NOT NULL,

relationship\_start DATE,

notes TEXT,

is\_primary\_doctor BOOLEAN DEFAULT FALSE,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (doctor\_id) REFERENCES customers(customer\_id),

FOREIGN KEY (patient\_id) REFERENCES customers(customer\_id)

);

**-- Cotizaciones**

CREATE TABLE quotes (

quote\_id INT AUTO\_INCREMENT PRIMARY KEY,

customer\_id INT NOT NULL,

doctor\_id INT,

quote\_number VARCHAR(20) UNIQUE NOT NULL,

validity\_days INT DEFAULT 30,

status ENUM('draft', 'sent', 'approved', 'rejected', 'expired', 'converted') DEFAULT 'draft',

subtotal DECIMAL(10,2) NOT NULL,

tax\_amount DECIMAL(10,2) NOT NULL,

discount\_amount DECIMAL(10,2) DEFAULT 0,

total\_amount DECIMAL(10,2) NOT NULL,

notes TEXT,

prescription\_id INT,

user\_id INT NOT NULL COMMENT 'User who created the quote',

expiration\_date DATE,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id),

FOREIGN KEY (doctor\_id) REFERENCES customers(customer\_id)

);

**-- Detalles de cotización**

CREATE TABLE quote\_items (

quote\_item\_id INT AUTO\_INCREMENT PRIMARY KEY,

quote\_id INT NOT NULL,

product\_id INT NOT NULL,

variant\_id INT,

quantity INT NOT NULL DEFAULT 1,

unit\_price DECIMAL(10,2) NOT NULL,

discount\_percent DECIMAL(5,2) DEFAULT 0,

tax\_percent DECIMAL(5,2) DEFAULT 0,

customization\_details TEXT,

notes TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (quote\_id) REFERENCES quotes(quote\_id),

FOREIGN KEY (product\_id) REFERENCES products(product\_id),

FOREIGN KEY (variant\_id) REFERENCES product\_variants(variant\_id)

);

**-- Órdenes de venta**

CREATE TABLE orders (

order\_id INT AUTO\_INCREMENT PRIMARY KEY,

customer\_id INT NOT NULL,

doctor\_id INT,

order\_number VARCHAR(20) UNIQUE NOT NULL,

order\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

quote\_id INT,

status ENUM('pending', 'processing', 'awaiting\_payment', 'awaiting\_customization',

'ready\_for\_delivery', 'delivered', 'completed', 'cancelled') DEFAULT 'pending',

payment\_status ENUM('pending', 'partial', 'paid', 'refunded') DEFAULT 'pending',

subtotal DECIMAL(10,2) NOT NULL,

tax\_amount DECIMAL(10,2) NOT NULL,

discount\_amount DECIMAL(10,2) DEFAULT 0,

shipping\_amount DECIMAL(10,2) DEFAULT 0,

total\_amount DECIMAL(10,2) NOT NULL,

prescription\_id INT,

shipping\_address TEXT,

billing\_address TEXT,

estimated\_delivery\_date DATE,

actual\_delivery\_date DATE,

notes TEXT,

user\_id INT NOT NULL COMMENT 'User who created the order',

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id),

FOREIGN KEY (doctor\_id) REFERENCES customers(customer\_id),

FOREIGN KEY (quote\_id) REFERENCES quotes(quote\_id)

);

**-- Detalles de orden**

CREATE TABLE order\_items (

order\_item\_id INT AUTO\_INCREMENT PRIMARY KEY,

order\_id INT NOT NULL,

product\_id INT NOT NULL,

variant\_id INT,

inventory\_id INT,

quantity INT NOT NULL DEFAULT 1,

unit\_price DECIMAL(10,2) NOT NULL,

discount\_percent DECIMAL(5,2) DEFAULT 0,

tax\_percent DECIMAL(5,2) DEFAULT 0,

status ENUM('pending', 'processing', 'ready', 'delivered', 'returned') DEFAULT 'pending',

customization\_details TEXT,

notes TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (order\_id) REFERENCES orders(order\_id),

FOREIGN KEY (product\_id) REFERENCES products(product\_id),

FOREIGN KEY (variant\_id) REFERENCES product\_variants(variant\_id),

FOREIGN KEY (inventory\_id) REFERENCES inventory\_items(inventory\_id)

);

**-- Prescripciones médicas**

CREATE TABLE prescriptions (

prescription\_id INT AUTO\_INCREMENT PRIMARY KEY,

doctor\_id INT NOT NULL,

patient\_id INT NOT NULL,

prescription\_date DATE NOT NULL,

diagnosis TEXT NOT NULL,

prescription\_details TEXT NOT NULL,

attachment\_url VARCHAR(255),

verification\_status ENUM('pending', 'verified', 'rejected') DEFAULT 'pending',

verified\_by INT,

verification\_date TIMESTAMP,

expiry\_date DATE,

notes TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (doctor\_id) REFERENCES customers(customer\_id),

FOREIGN KEY (patient\_id) REFERENCES customers(customer\_id)

);

**-- Pagos**

CREATE TABLE payments (

payment\_id INT AUTO\_INCREMENT PRIMARY KEY,

order\_id INT NOT NULL,

payment\_method ENUM('cash', 'credit\_card', 'bank\_transfer', 'insurance', 'check') NOT NULL,

amount DECIMAL(10,2) NOT NULL,

transaction\_reference VARCHAR(100),

payment\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

status ENUM('pending', 'completed', 'failed', 'refunded') DEFAULT 'pending',

gateway\_response TEXT,

notes TEXT,

created\_by INT NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (order\_id) REFERENCES orders(order\_id)

);

**-- Facturas**

CREATE TABLE invoices (

invoice\_id INT AUTO\_INCREMENT PRIMARY KEY,

order\_id INT NOT NULL,

invoice\_number VARCHAR(20) UNIQUE NOT NULL,

invoice\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

due\_date DATE,

customer\_id INT NOT NULL,

billing\_address TEXT,

subtotal DECIMAL(10,2) NOT NULL,

tax\_amount DECIMAL(10,2) NOT NULL,

discount\_amount DECIMAL(10,2) DEFAULT 0,

total\_amount DECIMAL(10,2) NOT NULL,

payment\_status ENUM('pending', 'partial', 'paid', 'cancelled') DEFAULT 'pending',

tax\_id VARCHAR(50),

notes TEXT,

created\_by INT NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (order\_id) REFERENCES orders(order\_id),

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id)

);

**Esquema de Tablas - Post-Venta y Devoluciones**

-- Seguimiento de adaptación

CREATE TABLE adaptation\_tracking (

tracking\_id INT AUTO\_INCREMENT PRIMARY KEY,

order\_item\_id INT NOT NULL,

patient\_id INT NOT NULL,

status ENUM('scheduled', 'in\_progress', 'completed', 'cancelled') DEFAULT 'scheduled',

start\_date DATE,

end\_date DATE,

follow\_up\_schedule TEXT,

adaptation\_notes TEXT,

clinical\_observations TEXT,

created\_by INT NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (order\_item\_id) REFERENCES order\_items(order\_item\_id),

FOREIGN KEY (patient\_id) REFERENCES customers(customer\_id)

);

**-- Citas de seguimiento**

CREATE TABLE follow\_up\_appointments (

appointment\_id INT AUTO\_INCREMENT PRIMARY KEY,

tracking\_id INT NOT NULL,

appointment\_date DATETIME NOT NULL,

appointment\_type ENUM('initial\_fitting', 'adjustment', 'check\_up', 'final\_review') NOT NULL,

status ENUM('scheduled', 'completed', 'cancelled', 'no\_show') DEFAULT 'scheduled',

doctor\_id INT,

location\_id INT,

notes TEXT,

outcome TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (tracking\_id) REFERENCES adaptation\_tracking(tracking\_id),

FOREIGN KEY (doctor\_id) REFERENCES customers(customer\_id),

FOREIGN KEY (location\_id) REFERENCES locations(location\_id)

);

**-- Solicitudes de servicio técnico**

CREATE TABLE service\_requests (

request\_id INT AUTO\_INCREMENT PRIMARY KEY,

customer\_id INT NOT NULL,

order\_item\_id INT NOT NULL,

request\_type ENUM('repair', 'adjustment', 'maintenance', 'inspection') NOT NULL,

priority ENUM('low', 'medium', 'high', 'urgent') DEFAULT 'medium',

status ENUM('submitted', 'evaluating', 'in\_progress', 'waiting\_parts', 'completed', 'cancelled') DEFAULT 'submitted',

description TEXT NOT NULL,

diagnostic TEXT,

solution TEXT,

request\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

assigned\_to INT,

completion\_date TIMESTAMP,

is\_warranty BOOLEAN DEFAULT FALSE,

estimated\_cost DECIMAL(10,2),

final\_cost DECIMAL(10,2),

notes TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id),

FOREIGN KEY (order\_item\_id) REFERENCES order\_items(order\_item\_id)

);

**-- Repuestos utilizados en servicios**

CREATE TABLE service\_parts (

service\_part\_id INT AUTO\_INCREMENT PRIMARY KEY,

request\_id INT NOT NULL,

product\_id INT NOT NULL,

variant\_id INT,

inventory\_id INT,

quantity INT NOT NULL DEFAULT 1,

unit\_cost DECIMAL(10,2) NOT NULL,

is\_charged BOOLEAN DEFAULT TRUE,

notes TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (request\_id) REFERENCES service\_requests(request\_id),

FOREIGN KEY (product\_id) REFERENCES products(product\_id),

FOREIGN KEY (variant\_id) REFERENCES product\_variants(variant\_id),

FOREIGN KEY (inventory\_id) REFERENCES inventory\_items(inventory\_id)

);

**-- Devoluciones**

CREATE TABLE returns (

return\_id INT AUTO\_INCREMENT PRIMARY KEY,

order\_id INT NOT NULL,

customer\_id INT NOT NULL,

return\_number VARCHAR(20) UNIQUE NOT NULL,

return\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

status ENUM('pending\_approval', 'approved', 'received', 'inspected', 'completed', 'rejected') DEFAULT 'pending\_approval',

return\_reason ENUM('defective', 'incorrect\_item', 'not\_as\_described', 'medical\_incompatibility', 'patient\_discomfort', 'other') NOT NULL,

detailed\_reason TEXT,

return\_type ENUM('refund', 'exchange', 'credit', 'repair') NOT NULL,

approval\_notes TEXT,

medical\_authorization BOOLEAN DEFAULT FALSE,

medical\_authorization\_by INT,

medical\_notes TEXT,

created\_by INT NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (order\_id) REFERENCES orders(order\_id),

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id),

FOREIGN KEY (medical\_authorization\_by) REFERENCES customers(customer\_id)

);

**-- Detalles de devolución**

CREATE TABLE return\_items (

return\_item\_id INT AUTO\_INCREMENT PRIMARY KEY,

return\_id INT NOT NULL,

order\_item\_id INT NOT NULL,

product\_id INT NOT NULL,

variant\_id INT,

quantity INT NOT NULL DEFAULT 1,

condition ENUM('new', 'used', 'damaged', 'non\_functional') NOT NULL,

restockable BOOLEAN DEFAULT FALSE,

inspection\_notes TEXT,

exchange\_product\_id INT,

exchange\_variant\_id INT,

refund\_amount DECIMAL(10,2),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (return\_id) REFERENCES returns(return\_id),

FOREIGN KEY (order\_item\_id) REFERENCES order\_items(order\_item\_id),

FOREIGN KEY (product\_id) REFERENCES products(product\_id),

FOREIGN KEY (variant\_id) REFERENCES product\_variants(variant\_id),

FOREIGN KEY (exchange\_product\_id) REFERENCES products(product\_id),

FOREIGN KEY (exchange\_variant\_id) REFERENCES product\_variants(variant\_id)

);

**-- Garantías**

CREATE TABLE warranties (

warranty\_id INT AUTO\_INCREMENT PRIMARY KEY,

product\_id INT NOT NULL,

warranty\_type ENUM('standard', 'extended', 'lifetime') DEFAULT 'standard',

duration\_months INT NOT NULL,

description TEXT,

conditions TEXT NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (product\_id) REFERENCES products(product\_id)

);

**-- Garantías aplicadas a productos vendidos**

CREATE TABLE product\_warranties (

product\_warranty\_id INT AUTO\_INCREMENT PRIMARY KEY,

order\_item\_id INT NOT NULL,

warranty\_id INT NOT NULL,

start\_date DATE NOT NULL,

end\_date DATE NOT NULL,

extended\_warranty\_sold BOOLEAN DEFAULT FALSE,

extended\_warranty\_price DECIMAL(10,2),

warranty\_certificate\_number VARCHAR(50),

notes TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (order\_item\_id) REFERENCES order\_items(order\_item\_id),

FOREIGN KEY (warranty\_id) REFERENCES warranties(warranty\_id)

);

**-- Disposición final de productos devueltos**

CREATE TABLE product\_dispositions (

disposition\_id INT AUTO\_INCREMENT PRIMARY KEY,

return\_item\_id INT NOT NULL,

disposition\_type ENUM('restocked', 'repaired', 'recycled', 'destroyed', 'returned\_to\_manufacturer') NOT NULL,

disposition\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

performed\_by INT NOT NULL,

location\_id INT,

destruction\_certificate VARCHAR(100),

notes TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (return\_item\_id) REFERENCES return\_items(return\_item\_id),

FOREIGN KEY (location\_id) REFERENCES locations(location\_id)

);

**TABLAS COMPLEMENTARIAS**

**-- Usuarios del sistema**

CREATE TABLE users (

user\_id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(50) UNIQUE NOT NULL,

password\_hash VARCHAR(255) NOT NULL,

first\_name VARCHAR(100) NOT NULL,

last\_name VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

phone VARCHAR(20),

role\_id INT NOT NULL,

location\_id INT,

last\_login TIMESTAMP,

status ENUM('active', 'inactive', 'suspended') DEFAULT 'active',

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (role\_id) REFERENCES roles(role\_id),

FOREIGN KEY (location\_id) REFERENCES locations(location\_id)

);

**-- Roles y permisos**

CREATE TABLE roles (

role\_id INT AUTO\_INCREMENT PRIMARY KEY,

role\_name VARCHAR(50) UNIQUE NOT NULL,

description TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

CREATE TABLE permissions (

permission\_id INT AUTO\_INCREMENT PRIMARY KEY,

permission\_name VARCHAR(100) UNIQUE NOT NULL,

description TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE role\_permissions (

role\_id INT NOT NULL,

permission\_id INT NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

PRIMARY KEY (role\_id, permission\_id),

FOREIGN KEY (role\_id) REFERENCES roles(role\_id),

FOREIGN KEY (permission\_id) REFERENCES permissions(permission\_id)

);

**-- Configuración del sistema**

CREATE TABLE system\_settings (

setting\_id INT AUTO\_INCREMENT PRIMARY KEY,

setting\_key VARCHAR(100) UNIQUE NOT NULL,

setting\_value TEXT,

setting\_type ENUM('text', 'number', 'boolean', 'json', 'date') DEFAULT 'text',

description TEXT,

is\_public BOOLEAN DEFAULT FALSE,

modified\_by INT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (modified\_by) REFERENCES users(user\_id)

);

**-- Log de actividad y auditoría**

CREATE TABLE activity\_logs (

log\_id INT AUTO\_INCREMENT PRIMARY KEY,

user\_id INT,

action\_type VARCHAR(50) NOT NULL,

entity\_type VARCHAR(50) NOT NULL,

entity\_id INT,

ip\_address VARCHAR(45),

user\_agent TEXT,

old\_values TEXT,

new\_values TEXT,

action\_timestamp TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (user\_id) REFERENCES users(user\_id)

);

**-- Encuestas de satisfacción**

CREATE TABLE satisfaction\_surveys (

survey\_id INT AUTO\_INCREMENT PRIMARY KEY,

customer\_id INT NOT NULL,

order\_id INT,

service\_request\_id INT,

survey\_type ENUM('post\_sale', 'post\_service', 'adaptation', 'general') NOT NULL,

satisfaction\_score INT,

product\_quality\_score INT,

service\_quality\_score INT,

would\_recommend BOOLEAN,

comments TEXT,

survey\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id),

FOREIGN KEY (order\_id) REFERENCES orders(order\_id),

FOREIGN KEY (service\_request\_id) REFERENCES service\_requests(request\_id)

);

**-- Documentos y archivos adjuntos**

CREATE TABLE documents (

document\_id INT AUTO\_INCREMENT PRIMARY KEY,

document\_type ENUM('prescription', 'invoice', 'certificate', 'technical\_sheet', 'warranty', 'return\_authorization', 'image', 'other') NOT NULL,

related\_entity\_type VARCHAR(50) NOT NULL,

related\_entity\_id INT NOT NULL,

file\_name VARCHAR(255) NOT NULL,

file\_path VARCHAR(255) NOT NULL,

file\_size INT,

file\_type VARCHAR(100),

upload\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

uploaded\_by INT,

description TEXT,

is\_confidential BOOLEAN DEFAULT FALSE,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (uploaded\_by) REFERENCES users(user\_id)

);

**-- Notificaciones**

CREATE TABLE notifications (

notification\_id INT AUTO\_INCREMENT PRIMARY KEY,

user\_id INT,

notification\_type VARCHAR(50) NOT NULL,

title VARCHAR(100) NOT NULL,

message TEXT NOT NULL,

related\_entity\_type VARCHAR(50),

related\_entity\_id INT,

is\_read BOOLEAN DEFAULT FALSE,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

read\_at TIMESTAMP,

FOREIGN KEY (user\_id) REFERENCES users(user\_id)

);

**ÍNDICES RECOMENDADOS**

Para optimizar el rendimiento de la base de datos, se recomienda implementar los siguientes índices:

-- Índices para búsquedas frecuentes

CREATE INDEX idx\_products\_category ON products(category\_id);

CREATE INDEX idx\_product\_variants\_product ON product\_variants(product\_id);

CREATE INDEX idx\_inventory\_items\_variant ON inventory\_items(variant\_id);

CREATE INDEX idx\_inventory\_items\_location ON inventory\_items(location\_id);

CREATE INDEX idx\_inventory\_items\_status ON inventory\_items(status);

CREATE INDEX idx\_inventory\_items\_expiration ON inventory\_items(expiration\_date);

CREATE INDEX idx\_orders\_customer ON orders(customer\_id);

CREATE INDEX idx\_orders\_status ON orders(status);

CREATE INDEX idx\_orders\_date ON orders(order\_date);

CREATE INDEX idx\_order\_items\_order ON order\_items(order\_id);

CREATE INDEX idx\_order\_items\_product ON order\_items(product\_id);

CREATE INDEX idx\_returns\_order ON returns(order\_id);

CREATE INDEX idx\_returns\_customer ON returns(customer\_id);

CREATE INDEX idx\_returns\_status ON returns(status);

CREATE INDEX idx\_return\_items\_return ON return\_items(return\_id);

CREATE INDEX idx\_service\_requests\_customer ON service\_requests(customer\_id);

CREATE INDEX idx\_service\_requests\_status ON service\_requests(status);

CREATE INDEX idx\_service\_requests\_priority ON service\_requests(priority);

CREATE INDEX idx\_customers\_type ON customers(customer\_type);

CREATE INDEX idx\_customers\_email ON customers(email);

CREATE INDEX idx\_customers\_name ON customers(last\_name, first\_name);

**-- Índices para relaciones frecuentes**

CREATE INDEX idx\_adaptation\_tracking\_order\_item ON adaptation\_tracking(order\_item\_id);

CREATE INDEX idx\_adaptation\_tracking\_patient ON adaptation\_tracking(patient\_id);

CREATE INDEX idx\_follow\_up\_appointments\_tracking ON follow\_up\_appointments(tracking\_id);

**-- Índices para búsquedas de texto**

CREATE FULLTEXT INDEX ft\_products\_name\_description ON products(name, description);

CREATE FULLTEXT INDEX ft\_customers\_name ON customers(first\_name, last\_name, company\_name);

**VISTAS RECOMENDADAS**

Para facilitar los reportes y consultas frecuentes, se recomienda implementar las siguientes vistas:

**-- Vista de inventario disponible**

CREATE VIEW vw\_available\_inventory AS

SELECT

p.product\_id,

p.name AS product\_name,

p.sku,

c.name AS category,

pv.variant\_id,

pv.size,

pv.color,

pv.material\_variant,

l.name AS location,

ii.inventory\_id,

ii.batch\_number,

ii.serial\_number,

ii.quantity,

ii.expiration\_date,

ii.unit\_cost

FROM

inventory\_items ii

JOIN product\_variants pv ON ii.variant\_id = pv.variant\_id

JOIN products p ON pv.product\_id = p.product\_id

JOIN categories c ON p.category\_id = c.category\_id

JOIN locations l ON ii.location\_id = l.location\_id

WHERE

ii.status = 'available'

AND ii.quantity > 0;

**-- Vista de productos próximos a vencer**

CREATE VIEW vw\_expiring\_products AS

SELECT

p.product\_id,

p.name AS product\_name,

p.sku,

pv.variant\_id,

pv.size,

pv.sku\_variant,

ii.inventory\_id,

ii.batch\_number,

ii.quantity,

ii.expiration\_date,

l.name AS location,

DATEDIFF(ii.expiration\_date, CURDATE()) AS days\_until\_expiration

FROM

inventory\_items ii

JOIN product\_variants pv ON ii.variant\_id = pv.variant\_id

JOIN products p ON pv.product\_id = p.product\_id

JOIN locations l ON ii.location\_id = l.location\_id

WHERE

ii.expiration\_date IS NOT NULL

AND ii.status = 'available'

AND ii.quantity > 0

AND ii.expiration\_date BETWEEN CURDATE() AND DATE\_ADD(CURDATE(), INTERVAL 90 DAY);

**-- Vista de ventas por período**

CREATE VIEW vw\_sales\_by\_period AS

SELECT

DATE\_FORMAT(o.order\_date, '%Y-%m') AS period,

COUNT(DISTINCT o.order\_id) AS total\_orders,

SUM(o.total\_amount) AS total\_sales,

AVG(o.total\_amount) AS average\_order\_value,

SUM(oi.quantity) AS total\_items\_sold

FROM

orders o

JOIN order\_items oi ON o.order\_id = oi.order\_id

WHERE

o.status NOT IN ('cancelled')

GROUP BY

DATE\_FORMAT(o.order\_date, '%Y-%m')

ORDER BY

period DESC;

**-- Vista de rendimiento de productos**

CREATE VIEW vw\_product\_performance AS

SELECT

p.product\_id,

p.name AS product\_name,

p.sku,

c.name AS category,

COUNT(DISTINCT oi.order\_item\_id) AS times\_sold,

SUM(oi.quantity) AS quantity\_sold,

SUM(oi.quantity \* oi.unit\_price) AS total\_revenue,

AVG(oi.unit\_price) AS average\_price,

(SELECT COUNT(\*) FROM return\_items ri JOIN order\_items oi2 ON ri.order\_item\_id = oi2.order\_item\_id

WHERE oi2.product\_id = p.product\_id) AS total\_returns,

(SELECT COUNT(\*) FROM service\_requests sr JOIN order\_items oi3 ON sr.order\_item\_id = oi3.order\_item\_id

WHERE oi3.product\_id = p.product\_id) AS service\_requests

FROM

products p

LEFT JOIN order\_items oi ON p.product\_id = oi.product\_id

LEFT JOIN orders o ON oi.order\_id = o.order\_id

JOIN categories c ON p.category\_id = c.category\_id

WHERE

o.status NOT IN ('cancelled')

OR o.status IS NULL

GROUP BY

p.product\_id, p.name, p.sku, c.name;

**-- Vista de garantías activas**

CREATE VIEW vw\_active\_warranties AS

SELECT

pw.product\_warranty\_id,

c.customer\_id,

CONCAT(c.first\_name, ' ', c.last\_name) AS customer\_name,

p.product\_id,

p.name AS product\_name,

pv.size,

oi.order\_item\_id,

o.order\_number,

o.order\_date,

pw.start\_date,

pw.end\_date,

w.warranty\_type,

w.duration\_months,

DATEDIFF(pw.end\_date, CURDATE()) AS days\_remaining

FROM

product\_warranties pw

JOIN order\_items oi ON pw.order\_item\_id = oi.order\_item\_id

JOIN orders o ON oi.order\_id = o.order\_id

JOIN customers c ON o.customer\_id = c.customer\_id

JOIN warranties w ON pw.warranty\_id = w.warranty\_id

JOIN products p ON oi.product\_id = p.product\_id

LEFT JOIN product\_variants pv ON oi.variant\_id = pv.variant\_id

WHERE

pw.end\_date >= CURDATE();

**CONSIDERACIONES DE IMPLEMENTACIÓN**

**Seguridad y Cumplimiento**

* Implementar cifrado de datos sensibles en la base de datos.
* Configurar la rotación automática de logs para evitar crecimiento excesivo.
* Establecer políticas de contraseñas robustas y autenticación de doble factor.
* Implementar auditoría completa para todas las operaciones críticas.
* Asegurar el cumplimiento de las normativas de datos médicos aplicables.

**Rendimiento**

* Utilizar particionamiento para tablas que crecen rápidamente (logs, movimientos de inventario).
* Configurar un esquema de respaldo que incluya respaldos incrementales diarios y completos semanales.
* Implementar replicación para alta disponibilidad en entornos de producción.
* Establecer tareas programadas para mantenimiento de índices y optimización de consultas.

**Escalabilidad**

* Diseñar para implementación con contenedores (Docker) para facilitar el despliegue y la escalabilidad.
* Implementar una arquitectura de microservicios para separar funcionalidades clave.
* Utilizar balanceo de carga para distribuir el tráfico en implementaciones de alto volumen.
* Considerar la implementación de capas de caché para mejorar el rendimiento de las consultas frecuentes.

**REQUISITOS DE INFRAESTRUCTURA RECOMENDADA**

**Entorno de Desarrollo**

* Servidor: 4 CPU, 8GB RAM
* Almacenamiento: 100GB SSD
* Base de Datos: MySQL 8.0 o MariaDB 10.5+
* Servidor Web: Nginx o Apache

**Entorno de Producción**

* Servidor de Aplicación: 8 CPU, 16GB RAM (escalable)
* Servidor de Base de Datos: 8 CPU, 32GB RAM
* Almacenamiento: 500GB SSD con capacidad de crecimiento
* Base de Datos: MySQL 8.0 con replicación
* Sistema de Respaldo: Respaldo automático con retención configurable
* Monitoreo: Implementación de herramientas como Prometheus/Grafana

**CONCLUSIÓN**

Este sistema de gestión de inventario de prótesis ha sido diseñado considerando las particularidades de este tipo de producto médico especializado, integrando no solo la gestión de inventario, sino también los procesos de venta, seguimiento post-venta y gestión de devoluciones bajo un marco regulatorio sanitario.

La estructura de la base de datos propuesta ofrece un balance entre normalización para mantener la integridad de los datos y practicidad para el rendimiento en operaciones cotidianas. Los índices y vistas recomendados facilitarán la generación de reportes y la optimización de consultas frecuentes.

Se recomienda implementar este sistema siguiendo las mejores prácticas de desarrollo de software actuales, incluyendo:

* Desarrollo basado en pruebas (TDD)
* Integración y despliegue continuos (CI/CD)
* Control de versiones riguroso
* Documentación exhaustiva de APIs y componentes
* Revisión regular de seguridad y rendimiento

La flexibilidad del diseño permitirá adaptaciones futuras para cumplir con cambios regulatorios o nuevos requerimientos del negocio.