

Aprendizaje-Express — ISS-08

Feature Sale + ProductSale (Ventas y Transacciones ACID)

Maestro-Detalle • Unit of Work • Bloqueo Pesimista (FOR UPDATE) • Control de Stock

Propósito del ISS-o8: Gestión Atómica de Ventas

Relación Maestro-Detalle

Registra la cabecera de venta (Sale) vinculada a múltiples líneas de detalle (ProductSale) preservando el importe histórico.

Transacciones ACID

Garantiza el patrón Unit of Work: si falla el stock o el detalle, se ejecuta ROLLBACK completo sin dejar registros huérfanos.

Bloqueo Pesimista (FOR UPDATE)

Evita condiciones de carrera en inventario bloqueando la fila del producto antes de verificar y descontar stock.

Integración en /api/ventas

Expone la API completa de ventas en modalidad SIN AUTH con DTOs de creación transaccional, seeder y Swagger.

Modelo de Datos: Sales & ProductSales

Cabecera: Sale (sales)

- client_id (FK -> clients.id): Cliente comprador
- total_amount: Importe total recalculado
- sale_date: Fecha y hora de la transacción
- status: 'completed' | 'cancelled' | 'inactive'
- Relaciones: belongsTo(Client), hasMany(ProductSale)

Detalle: ProductSale (product_sales)

- sale_id (FK -> sales.id): Cabecera vinculada
- product_id (FK -> products.id): Producto vendido
- quantity: Cantidad comprada
- unit_price: SNAPSHOT del precio al vender
- subtotal: $\text{quantity} * \text{unit_price}$
- Relaciones: belongsTo(Sale), belongsTo(Product)

Secuencia Transaccional: withTransaction & Lock

1. Iniciar Transacción

withTransaction crea el scope aislado en la BD.

2. Validar Cliente

Verifica existencia y estado activo del cliente.

3. Bloquear Producto

SELECT ... FOR UPDATE evita escrituras concurrentes.

4. Verificar Stock

Valida stock $\geq$ quantity. Si no alcanza -> ROLLBACK.

5. Descontar Stock

Actualiza stock del producto dentro de la transacción.

6. Registrar Venta

Crea Sale + ProductSales y ejecuta COMMIT final.

Regla de Negocio: Snapshot de Precio unit_price

¿Por qué duplicar unit_price en ProductSale si el precio ya existe en Product?

1. Inmutabilidad Histórica: Si el precio del producto cambia mañana de \$10.00 a \$15.00, las ventas pasadas deben conservar \$10.00.
2. Auditoría Financiera: El total de la venta debe coincidir exactamente con la suma de los subtotales registrados en el momento de la compra.
3. Captura Automática: El servicio consulta el precio actual del producto durante la transacción e inyecta ese valor en unit_price de ProductSale.
4. Cálculo de Subtotal: $\text{subtotal} = \text{quantity} * \text{unit_price}$ (calculado de forma consistente en el backend).

Cableado de Asociaciones: sales.associations.ts

Relaciones Sequelize Declaradas Explícitamente

- `Sale.belongsTo(Client, { foreignKey: 'client_id', as: 'client' })`
- `Client.hasMany(Sale, { foreignKey: 'client_id', as: 'sales' })`
- `Sale.hasMany(ProductSale, { foreignKey: 'sale_id', as: 'items' })`
- `ProductSale.belongsTo(Sale, { foreignKey: 'sale_id', as: 'sale' })`
- `ProductSale.belongsTo(Product, { foreignKey: 'product_id', as: 'product' })`
- Importación con efecto secundario: `import './features/business/sales/sales.associations';`

Seeder de Ventas y Especificación Swagger

Seeder: `sales.seeder.ts`

- Genera ventas de prueba seleccionando clientes y productos activos.
- Descuenta stock real y calcula importes dinámicos con Faker.
- Integrado en SeedersRunner (`counts.ts: --sales=N`).
- Garantiza idempotencia truncando tablas relacionadas en orden.

Swagger: `sales.swagger.ts`

- Tag: Ventas (SIN AUTH).
- Endpoints: POST `/api/ventas`, GET `/api/ventas`, GET `/api/ventas/{id}`.
- Esquemas OpenAPI para `CreateSaleDto` y `SaleResponseDto` con anidamiento de líneas.
- Registrado centralmente en `src/swagger/index.ts`.

Criterios GATE — Definición de Terminado (DoD)

Checklist de Cierre de ISS-o8:

- ✓ Compilación de tipos limpia: `npx tsc --noEmit` finaliza con 0 errores.
- ✓ Tablas sincronizadas: `sales` y `product_sales` creadas físicamente en la BD.
- ✓ Transaccionalidad probada: intent de venta sin stock ejecuta ROLLBACK y retorna 400.
- ✓ Persistencia de histórico: `unit_price` preserva el snapshot del producto.
- ✓ Seeder funcional: `npm run db:seed -- --sales=10` puebla ventas de prueba correctamente.
- ✓ OpenAPI/Swagger: Documentación visible e interactiva bajo la etiqueta Ventas (SIN AUTH).
- ✓ Habilita Cierre de Fase I: Cierra las 5 features de negocio antes de introducir la seguridad RBAC (ISS-09).