UNIVERSIDAD TECNOLOGICA NACIONAL

Tecnicatura Universitaria en Programación - Universidad Tecnológica Nacional

TRABAJO PRÁCTICO INTEGRADOR

Programación II

Alumno:

Gomez Saucedo Augusto Nahuel - gomeznahuel@hotmail.com

1. Integrante y rol

Alumno: Gomez Saucedo Augusto Nahuel

Rol asumido:

- Diseño del modelo de dominio
- Implementación de DAO y Service
- Implementación de transacciones
- Diseño de base de datos MySQL
- Pruebas funcionales y documentación

2. Elección del dominio

Se eligió el dominio **Producto → CodigoBarras** por tratarse de una relación clara de cardinalidad 1 a 1, donde cada producto posee exactamente un código de barras asociado.

Este dominio permite aplicar:

- Integridad referencial
- Restricciones de unicidad
- Validaciones de dominio
- Manejo transaccional
- Arquitectura por capas

3. Diseño y modelo UML

Se diseñó una relación unidireccional 1 a 1 desde la clase `Producto` hacia `CodigoBarras`.

La cardinalidad se garantiza en base de datos mediante:

- Clave foránea `producto_id`

- Restricción `UNIQUE` sobre `producto_id`
- `ON DELETE CASCADE`

Esto asegura que:

- Un producto tiene un único código
- Un código pertenece a un único producto
- No existen registros huérfanos

(Insertar aquí la imagen UML)

4. Arquitectura del proyecto

El proyecto fue organizado en capas:

config/

Contiene la clase `DbConfig`, responsable de gestionar la conexión a la base de datos.

entities/

Define las clases del dominio:

- Producto
- CodigoBarras

Ambas incluyen:

- id (clave primaria)
- eliminado (baja lógica)

dao/

Contiene:

- GenericDao<T>
- ProductoDao
- CodigoBarrasDao

Se utilizan exclusivamente `PreparedStatement` para prevenir inyección SQL.

service/

Contiene la lógica de negocio y orquesta transacciones.

El `ProductoService`:

- Inicia transacciones
- Ejecuta operaciones compuestas
- Realiza commit o rollback según resultado
- Valida reglas de negocio

main/

Contiene `App`, que implementa el menú por consola.

5. Persistencia y base de datos

Se creó la base `tfi_p2_bd1` con dos tablas:

producto

- id BIGINT PK
- nombre NOT NULL
- precio DECIMAL CHECK (precio > 0)
- peso CHECK (NULL o > 0)

- eliminado BOOLEAN

codigo_barras

- id BIGINT PK
- valor UNIQUE
- producto_id UNIQUE FK
- tipo ENUM

La relación 1 a 1 se garantiza mediante:

UNIQUE (producto_id)

FOREIGN KEY (producto_id) REFERENCES producto(id)

ON DELETE CASCADE

6. Manejo de transacciones

Las operaciones críticas se realizan con:

connection.setAutoCommit(false);

En caso de éxito:

connection.commit();

En caso de error:

connection.rollback();

Esto garantiza consistencia ante fallos parciales.

7. Validaciones implementadas

- precio > 0
- peso NULL o > 0

- valor de código único
- producto_id único
- baja lógica (eliminado)

8. Pruebas realizadas

Se probaron:

- Creación de producto con código
- Actualización
- Eliminación lógica
- Violación de UNIQUE
- Violación de CHECK
- Rollback ante error simulado

Las capturas se incluyen en el informe.

9. Conclusiones

El proyecto permitió integrar:

- Arquitectura en capas
- Patrón DAO
- JDBC con PreparedStatement
- Manejo real de transacciones
- Modelado 1 a 1 consistente en código y base de datos

Se logró un sistema robusto, seguro y coherente entre modelo conceptual, implementación y persistencia.