Informe de trabajo práctico integrador de programación 2.

El proyecto consiste en el desarrollo de un sistema para la gestión de Libros y sus respectivas Fichas Bibliográficas, aplicando arquitectura por capas, relaciones 1→1, validaciones y persistencia con base de datos.

Integrantes: Gino Canevaro – Comisión 6 / Daiana Ghisio – Comisión 7 / Nicolás Otaño – Comisión 14 / Araceli Cabañas – Comisión 5.

ROLES: El desarrollo del trabajo práctico se llevó a cabo de manera colaborativa, asignando a cada integrante un rol específico. La división de tareas permitió organizar el trabajo de forma eficiente y asegurar consistencia entre las distintas capas del proyecto.

Integrante 1 (Gino Canevaro): Desarrollo de DAOs (Data Access Objects).

Fue responsable de la capa de persistencia del sistema.

- Manejo de conexiones JDBC, consultas SQL y transacciones.
- Aplicación del patrón DAO para garantizar independencia entre la lógica de negocio y la base de datos.
- Implementación del soft delete, claves foráneas y operaciones asociadas a la relación 1→1 entre libro y ficha.

Integrante 2 (Daiana Ghisio): Desarrollo del Main y Menú del Sistema.

Encargada del funcionamiento del menú principal orientado al dominio.

- Construcción de la interfaz por consola para gestionar Libros y Fichas Bibliográficas.
- Navegación entre operaciones CRUD para cada entidad.
- Validación básica de ingreso de datos y comunicación con la capa de servicios.
- Integración de las diferentes funciones del sistema desde la perspectiva del usuario.

Integrante 3 (Nicolás Otaño):

Responsable de la capa de servicios del dominio.

- Validación del año de publicación

- Control de unicidad de la ficha por libro.
- Prevención de inconsistencias en la relación 1→1.
- Coordinación entre DAOs y la interfaz del sistema.
- Manejo de errores y validaciones previas a persistir cambios

Integrante 4 (Araceli Cabañas): Modelos del Dominio (Models) y Diagrama UML

Responsable del diseño conceptual y estructural del sistema.

- Modelado de las entidades Libro, FichaBibliografica y la clase abstracta Base.
- Definición de atributos, constructores, validaciones y relaciones.
- Elaboración del diagrama UML de clases, reflejando herencia, dependencias y la relación 1→1 entre las entidades.
- Diseño coherente paya mayor facilidad del trabajo de las capas DAO y Service.

Elección del Dominio y Justificación:

Para el desarrollo del Trabajo Práctico se seleccionó el dominio “Gestión de Libros y Fichas Bibliográficas”, compuesto por dos entidades principales:

- Libro
- FichaBibliografica

La elección de este dominio se fundamenta en varios motivos técnicos y pedagógicos:

1. Claridad conceptual y relevancia académica

2. Presencia de relaciones interesantes para modelar

3. Complejidad suficiente para arquitectura por capas

4. Aplicación de reglas de negocio reales

5. Facilita el uso de transacciones y operaciones atómicas

6. Posibilidad de extenderlo en futuras versiones

Diseño:

Tipo de relación elegida: 1 → 1

La relación entre Libro y FichaBibliografica es uno a uno porque:

Un Libro puede tener opcionalmente una única FichaBibliografica, la cual pertenece exclusivamente a ese Libro, garantizando integridad y evitando duplicaciones.

Dirección de la relación: 1 → 1 unidireccional

La dirección elegida es:

FichaBibliografica → Libro

Es decir:

- `FichaBibliografica` contiene una referencia a `Libro`.
- `Libro` no contiene una referencia de vuelta.
- La navegación es solo en un sentido.

Justificación:

La ficha se construye a partir de los datos del libro y se accede naturalmente desde ella a su Libro asociado, evitando dependencias cíclicas y facilitando la serialización, el mapeo y las pruebas unitarias.

Implementación física: FK única

Se utilizó clave foránea única (id_libro UNIQUE) dentro de `FichaBibliografica`

Esto significa que:

La ficha referencia al libro mediante una FK con restricción UNIQUE para asegurar una única ficha por libro, manteniendo claves primarias independientes en ambas tablas.

Justificación de FK única sobre PK compartida:

Ventajas de FK única

- Mantiene independencia total entre Libro y Ficha.

- Permite crear libros sin ficha.
- Evita acoplamiento innecesario.
- Modelo más simple de mantener.

Desventajas de PK compartida

- Obliga a crear ambas entidades simultáneamente.
- Reduce flexibilidad.
- Aumenta el acoplamiento en el dominio.
- Difícil de serializar o mapear vía DAOs.

Relación final UML:

- Unidireccional
- Flecha apuntando desde FichaBibliografica → Libro
- Multiplicidad 1 → 1
- Implementada con FK única id_libro

Arquitectura por Capas:

El sistema está desarrollado utilizando una arquitectura por capas, lo que permite separar responsabilidades, mantener el código ordenado y facilitar la escalabilidad. Las capas utilizadas son las siguientes:

1. Capa de Presentación (UI / Main)

Ubicación: `TPI_grupo94.Main`

Es la parte del sistema con la que interactúa el usuario mediante consola.
Incluye:

- AppMenu: inicia el sistema, muestra el menú y procesa opciones.
- MenuHandler: ejecuta las operaciones seleccionadas.
- MenuDisplay: imprime los menús en pantalla.
- TestConexion: clase auxiliar para probar la conexión a la base de datos.

Responsabilidad: recibir entrada del usuario, mostrar resultados y delegar operaciones a la capa de servicios.

2. Capa de Servicio (Lógica de Negocio)

Ubicación: `TPI_grupo94.Service`

Incluye:

- LibroServiceImpl
- FichaBibliograficaServiceImpl

Estas clases contienen la lógica de negocio, coordinando acciones entre la UI y los DAOs.

Responsabilidades:

- Validar datos.
- Gestionar reglas del negocio (relación 1:1 entre Libro y FichaBibliografica).
- Encapsular operaciones complejas.
- Gestionar transacciones cuando es necesario.

3. Capa de Acceso a Datos (DAO)

Ubicación: `TPI_grupo94.Dao`

Incluye:

- GenericDAO<T> (interfaz general)
- LibroDAO
- FichaBibliograficaDAO

Cada DAO se encarga de interactuar directamente con la base de datos utilizando JDBC.

Responsabilidades:

- Ejecutar operaciones SQL (INSERT, SELECT, UPDATE, soft delete).
- Mapear filas de la base de datos a objetos Java.
- Mantener la persistencia de las entidades.
- Aplicar el patrón Soft Delete mediante el campo `eliminado`.

4. Capa de Modelo (Entidades del Dominio)

Ubicación: `TPI_grupo94.Models`

Incluye:

- Base (superclase con id y eliminado)
- Libro
- FichaBibliografica

Representan las tablas de la base de datos y mantienen la estructura del dominio.

Responsabilidades:

- Representar datos del sistema.
- Facilitar el intercambio de información entre capas.
- Mantener las relaciones (Libro ↔ FichaBibliografica).

Persistencia:

La persistencia del sistema se implementa mediante el patrón DAO (Data Access Object) y el uso directo de JDBC para interactuar con la base de datos.
El objetivo principal es separar la lógica de acceso a datos del resto del sistema, logrando un código más organizado y fácil de mantener.

Transacciones:

Se usan commit/rollback especialmente para:

- Insertar libro + ficha
- Actualizaciones que afectan ambas entidades
  Flujo:

1. `setAutoCommit(false)`
2. Ejecutar operaciones
3. `commit()` si todo funciona
4. `rollback()` si hay errores

Validaciones y Reglas de Negocio:

- Año permitido: 1450–2025
- Solo una ficha por libro
- No permitir crear una ficha sin libro asociado
- Evitar duplicación de relaciones 1→1

Pruebas realizadas:

- Pruebas de menú: altas, bajas lógicas, modificaciones y listados.
- Consultas SQL en BD verificando:
    - Soft delete correcto
    - Relación 1→1 sin duplicados
    - Restricciones (CHECK, UNIQUE, FK)

Conclusiones y Mejoras Futuras:

El sistema cumple los objetivos académicos al implementar una relación 1→1, una arquitectura por capas y el uso de transacciones. Se proponen mejoras como una interfaz gráfica, búsquedas por autor o título y un sistema más avanzado de logs y manejo de errores

Fuentes y Herramientas:

- JDK + NetBeans
- JDBC + MySQL
- GitHub
- Documentación oficial de Java y JDBC
- Material de cátedra