TECNICATURA UNIVERSITARIA
EN PROGRAMACIÓN
A DISTANCIA

UTN
UNIVERSIDAD TECNOLÓGICA NACIONAL

Trabajo Práctico Integrador - Programación II

Grupo 6

Aplicación Java – Relación 1:1 Empleado → Legajo

Alumnos:
Agustin Hurtado
Alejandro Pedrosa
Luciano De La Rubia
Bruno Pighin

Introducción

El presente informe documenta el desarrollo completo del Trabajo Final Integrador para la cátedra Programación 2 de la Universidad Tecnológica Nacional. El objetivo principal es diseñar, implementar y documentar una aplicación Java basada en una relación 1 a 1 unidireccional, incorporando principios de arquitectura en capas, acceso a datos mediante JDBC y manejo transaccional.

Integrantes del equipo y roles

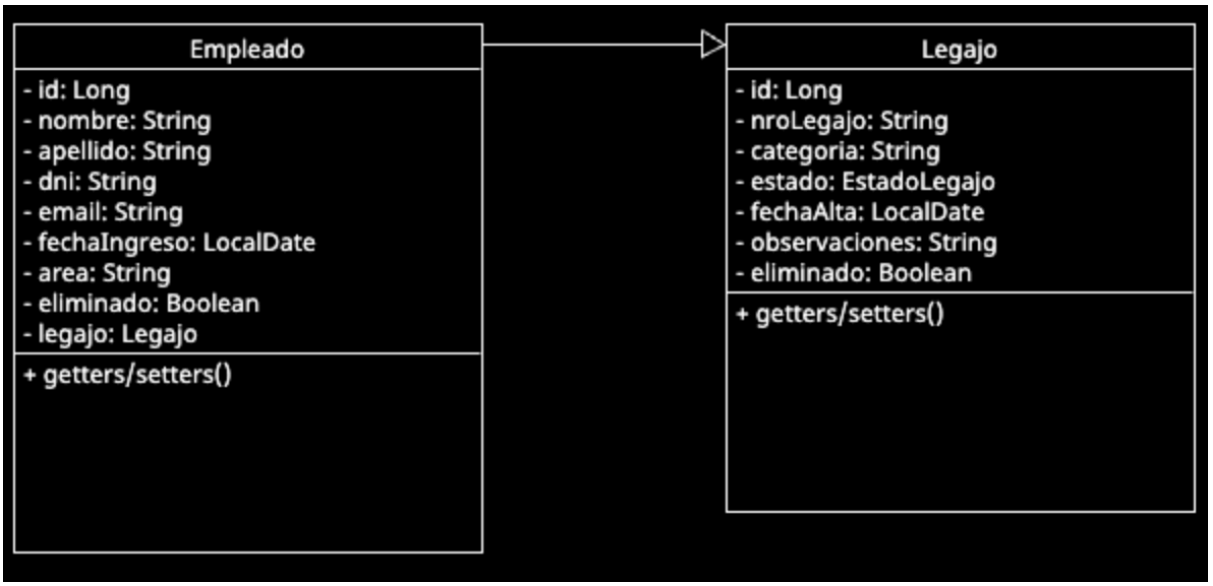
Alejandro Pedrosa: Desarrollo general, lógica de negocio, manejo transaccional .
Agustin Hurtado: Diseño del modelo entidad-relación, scripts SQL y pruebas sobre base de datos.
Luciano de la Rubia: Desarrollo del menú, interacción con usuario y manejo de errores
Bruno Pighin: Testing global, documentación formal e integración del video de presentación

Dominio elegido y justificación

El dominio seleccionado fue Empleado → Legajo, en el cual cada empleado posee un único legajo administrativo. Esta elección se fundamenta en los siguientes criterios:
• Es un caso común en sistemas de gestión de recursos humanos.
• Representa claramente una relación 1 a 1 unidireccional.
• Permite implementar validaciones reales (DNI único, nroLegajo único).
• Facilita la aplicación de transacciones (empleado + legajo como operación indivisible).

Análisis y diseño

UML del sistema



El diseño incluye las entidades Empleado y Legajo, representando la relación 1→1 donde Empleado contiene una referencia directa a Legajo.

Decisiones estructurales

Se adoptó una arquitectura en capas para asegurar legibilidad, mantenibilidad y una clara separación de responsabilidades, tal y como sugieren en las consignas del trabajo práctico integrador:

- Capa Model: entidades de dominio
- Capa DAO: interacción directa con la base de datos
- Capa Service: lógica de negocio y coordinación de transacciones
- Capa Config: manejo de la conexión a MySQL
- Capa Presentación: menú interactivo por consola

Implementación técnica

Capa Model

Incluye las clases Empleado y Legajo, ambas derivadas conceptualmente de una clase Base qué provee atributos comunes como id y eliminado.

Capa DAO

La capa DAO implementa acceso a datos mediante JDBC y PreparedStatement.
Funciones implementadas:

- insertar
- insertar transaccional
- actualizar
- eliminar lógico
- getById
- getAll

Además, se utilizan métodos con Connection externa para participar en transacciones.

Capa Service

Responsable de:

- validaciones
- reglas de negocio
- manejo de errores
- coordinación de transacciones

La inserción de un empleado con su legajo es completamente atómica.

Base de datos y persistencia

Modelo relacional

Se definieron dos tablas:

• empleados

• legajos

La relación 1⟶1 se garantiza mediante:

empleado_id BIGINT NOT NULL UNIQUE

Scripts SQL

Se incluyen dos archivos:

• 01_create_empresa_empleados.sql

• 02_seed_empresa_empleados.sql

Ambos probados en MySQL y compatibles con XAMPP y MySQL Workbench.

Conexión JDBC

Se utiliza DriverManager y MySQL Connector/J.

La clase DatabaseConnection valida la configuración de forma temprana (fail-fast).

Manejo de transacciones

La operación de crear un empleado junto con su legajo se realiza dentro de una misma transacción:

```
try {
conn.setAutoCommit(false);
insertar empleado;
insertar legajo;
conn.commit();
} catch (Exception e) {
conn.rollback();
}
```

Esto asegura:

• atomicidad

• integridad de datos

• aislamiento frente a errores

Validaciones

La aplicación implementa un conjunto de validaciones estrictas en la capa **Service**, cuyo propósito es garantizar la integridad de los datos antes de realizar cualquier operación de persistencia. Este enfoque evita inconsistencias tanto en la lógica del sistema como en la base de datos, asegurando que únicamente se procesen registros correctos y completos. Las validaciones se ejecutan de manera previa al acceso a la capa DAO, respetando el principio de separación de responsabilidades.

Validaciones en la entidad Empleado

Las reglas aplicadas a la entidad *Empleado* aseguran que los datos esenciales se encuentren presentes y sean coherentes:

Campos obligatorios

- **Nombre**: no puede ser nulo ni estar vacío, ya que identifica al empleado.
- **Apellido**: se requiere por las mismas razones que el nombre.
- **DNI**: obligatorio y utilizado como criterio principal de identificación.
  Fecha de ingreso: no puede ser nula, ya que se utiliza para determinar la antigüedad laboral.

Unicidad del DNI

Antes de realizar una operación de inserción o actualización, el sistema verifica si el DNI ya se encuentra registrado:

- Si el DNI pertenece a otro empleado distinto, la operación se rechaza.

- En actualizaciones, se permite únicamente si corresponde al mismo empleado.

Esta validación evita duplicar la identidad de un empleado dentro del sistema.

Validación del email

El correo electrónico es opcional, pero cuando se ingresa se valida mediante una expresión regular que verifica un formato básico correcto (*usuario@dominio*).
Esto previene registros con direcciones mal formadas o incorrectas.

Baja lógica

La eliminación de un empleado se implementa mediante una baja lógica utilizando el campo:

```
eliminado = TRUE
```

Este mecanismo permite ocultar registros sin eliminarlos físicamente, preservando información relevante para auditorías y evitando la pérdida definitiva de datos.

Validaciones en la entidad Legajo

Dado que el legajo constituye un documento administrativo único por empleado, se aplican las siguientes validaciones:

Campos obligatorios

- **Número de legajo**: requerido y con restricción de unicidad.
- **Estado**: debe corresponder a uno de los valores definidos en el tipo enumerado (*ACTIVO* o *INACTIVO*).
- **Fecha de alta**: no puede ser nula, dado que representa el momento de creación del legajo.

Unicidad del número de legajo

Previo a insertar o modificar un registro, se comprueba que no exista otro legajo con el mismo número.
Esta restricción se controla tanto en la capa Service como mediante una constraint UNIQUE en la base de datos.

Vinculación obligatoria

Cada legajo debe estar asociado a un empleado mediante la clave foránea:

```
empleado_id BIGINT NOT NULL UNIQUE
```

La restricción UNIQUE garantiza la relación **uno a uno**, evitando que un empleado tenga más de un legajo o que un legajo quede asociado a más de un empleado.

Validaciones en operaciones transaccionales

Cuando el legajo se crea conjuntamente con el empleado, ambas inserciones se realizan dentro de una misma transacción:

- Si el legajo no supera sus validaciones, se lanza una excepción y se produce un *rollback*.
- Esto asegura que el empleado no quede registrado sin su legajo en operaciones que deben ser atómicas.

Validaciones compartidas y consistencia global

Manejo de errores y mensajes claros

Todas las validaciones generan excepciones controladas con mensajes explicativos, facilitando la identificación del error y asegurando una retroalimentación clara hacia el usuario.

Separación adecuada de responsabilidades

La capa Service centraliza la lógica de negocio y las validaciones, mientras que la capa DAO se encarga únicamente del acceso a datos.
Este diseño garantiza claridad, mantenimiento sencillo y cumplimiento de buenas prácticas.

Prevención de inconsistencias

El uso de transacciones garantiza que:

- Las operaciones críticas (como crear empleado + legajo) sean completamente atómicas.
- Cualquier fallo en una de las operaciones implique revertir todo el proceso.

Este esquema mantiene la coherencia del sistema y evita estados intermedios inválidos.

Pruebas realizadas

Se realizaron pruebas de:
• Inserción simple y con legajo
• Listado de empleados y legajos
• Actualización de campos
• Eliminación lógica
• Rollback ante errores inducidos

• Búsqueda por filtros (nombre/apellido/DNI)
• Integridad relacional entre tablas
Todas las pruebas se realizaron exitosamente con capturas disponibles para el informe final.

Conclusiones

El desarrollo de este Trabajo Final Integrador permitió aplicar de manera práctica los conceptos fundamentales de Programación 2 y Bases de Datos, integrando programación orientada a objetos, arquitectura en capas y acceso a bases de datos mediante JDBC. La implementación del sistema Empleado–Legajo demostró correctamente la relación 1 a 1 unidireccional requerida, asegurando integridad tanto en el modelo como en la base de datos.

La utilización del patrón DAO, junto con la capa Service para validaciones y manejo transaccional, permitió construir una solución ordenada, modular y robusta. La creación transaccional de empleado y legajo garantizó atomicidad y consistencia ante posibles errores. Además, se cumplieron todas las funcionalidades solicitadas: CRUD completos, baja lógica, búsquedas relevantes, scripts SQL reproducibles y un menú interactivo funcional.

En síntesis, el trabajo consolida los conocimientos adquiridos durante la cursada y dio lugar a una aplicación estable y extensible, que puede servir como base para futuras mejoras como interfaz gráfica, uso de ORM o ampliación del dominio.