PROYECTO U3. Herramientas de mapeo objeto-relacional (ORM)

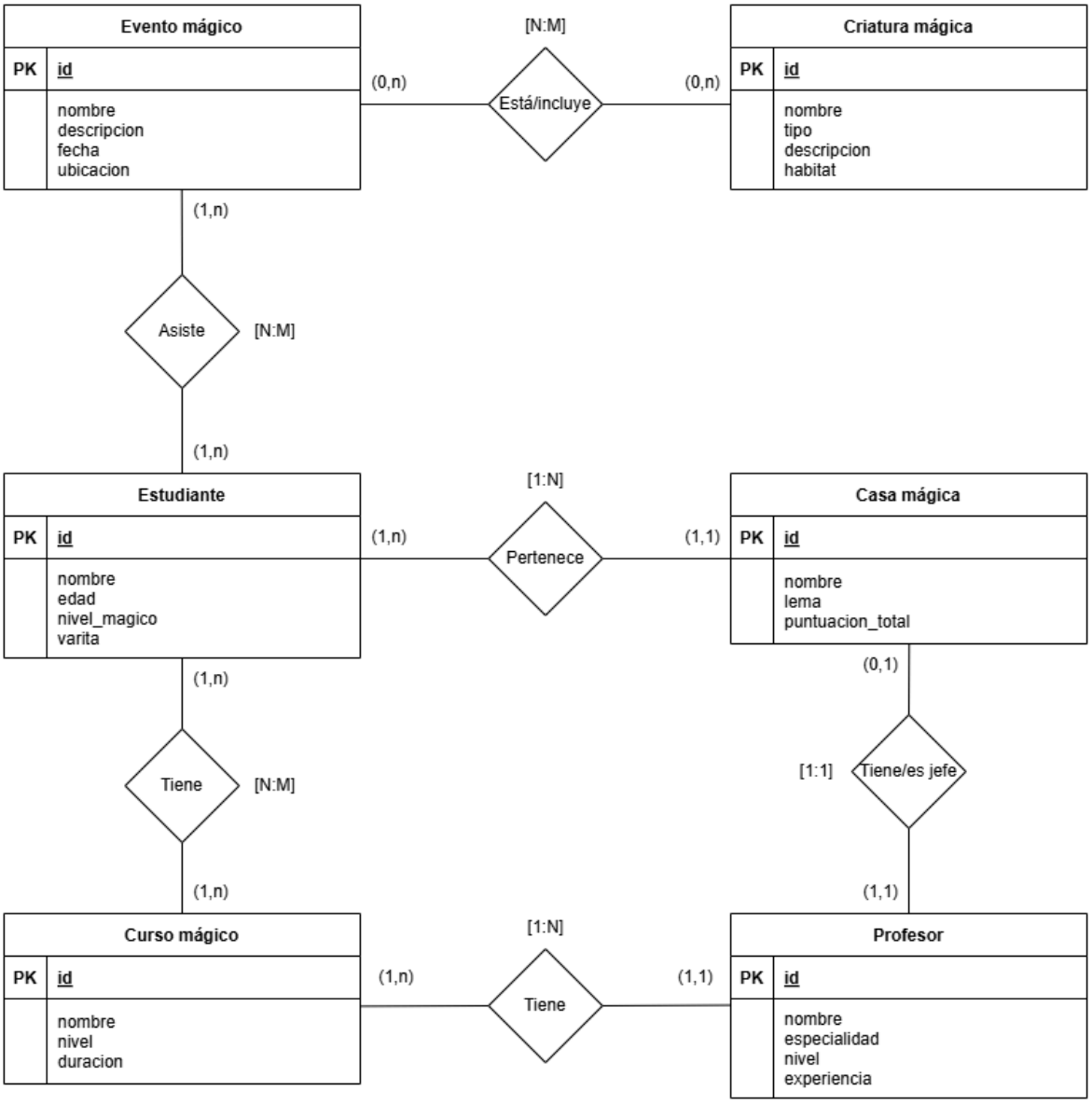
Sprint 1 Demo day 2/12/24

Desarrolla una aplicación con Hibernate que permita gestionar los datos de una base de datos concreta diferente de la que hemos trabajado en clase.

Se pide **REALIZAR y EXPLICAR** cada uno de los puntos siguientes:

1. Define el objetivo de tu sistema: qué gestiona en una dos o tres líneas.
   Sistema de Gestión de Hogwarts (Escuela de Magia y Hechicería).
   Objetivo del sistema: El sistema permite gestionar estudiantes, cursos, profesores, casas mágicas, criaturas y eventos mágicos de una academia.

2. Define el modelo de datos y sus relaciones. Al menos tienes que tener 6 entidades con sus correspondientes atributos. El sistema deberá contener, como mínimo una relación de cada tipo: 1:1, 1:N unidireccional y otra bidireccional, N:M.



Criatura mágica tiene una relación muchos a muchos con evento mágico porque una criatura puede no pertenecer a ningún evento o a muchos eventos. Del mismo modo un evento puede incluir ninguna o muchas criaturas.

Evento mágico tiene una relación muchos a muchos con estudiante porque un evento puede incluir a uno o más estudiantes y un estudiante puede asistir a uno o más eventos.

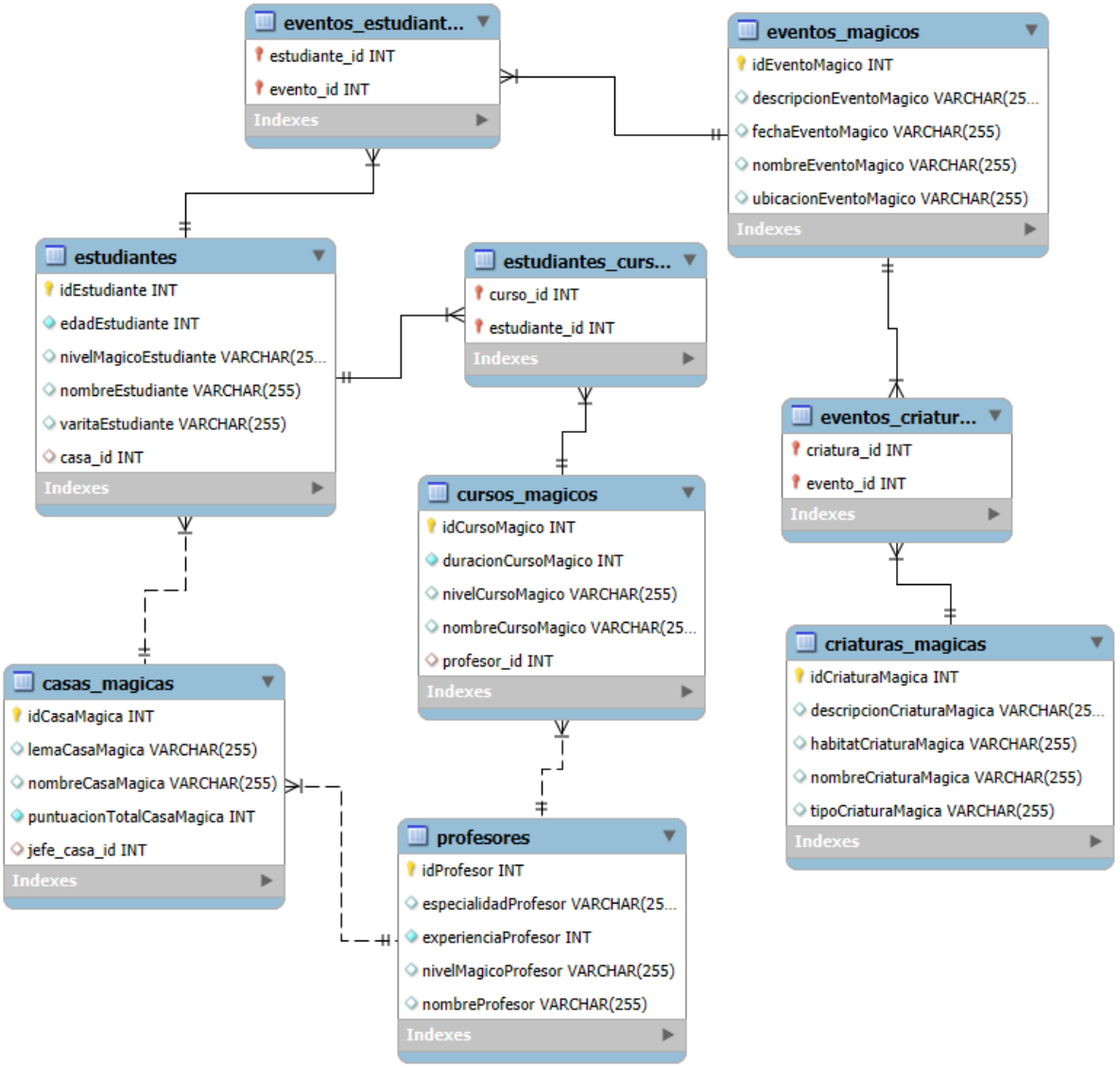
Estudiante tiene una relación uno a muchos con casa mágica porque un estudiante pertenece a una única casa y una casa tiene uno o muchos estudiantes.

Estudiante tiene una relación muchos a muchos con curso mágico porque un estudiante tiene uno o más cursos y un curso tiene a uno o más estudiantes.

Casa mágica tiene una relación uno a uno con profesor porque una casa tiene un único profesor jefe de la casa y un profesor puede ser jefe (o no) de alguna casa.

Profesor tiene una relación uno a muchos con curso mágico porque un profesor puede tener uno o más cursos y un curso tiene un único profesor.

3. Define su Esquema del modelo relacional de la BD escogida y el usuario.



Tablas principales

1. estudiantes:
   Contiene datos de los estudiantes, como su ID, edad, nivel mágico, nombre, varita y a qué casa mágica pertenecen (relación con casas_magicas).

❖ Relación:
   - ➢ Se relaciona con casas_magicas mediante la clave foránea casa_id.
   - ➢ Tiene una relación con los cursos (estudiantes_cursos).
   - ➢ También está asociada con eventos mediante eventos_estudiantes.

2. casas_magicas:
   Representa las casas mágicas a las que pertenecen los estudiantes.
   Campos: ID, lema, nombre, puntuación total y el ID del jefe de la casa.

❖ Relación:
   - ➢ Se conecta con estudiantes (una casa puede tener muchos estudiantes).
   - ➢ El campo jefe_casa_id hace referencia a un profesor que actúa como jefe de la casa (relación con profesores).

3. profesores:
   Guarda información sobre los profesores, como su ID, especialidad, experiencia, nivel mágico y nombre.

❖ Relación:
   - ➢ Está relacionado con cursos_magicos, ya que un profesor puede impartir cursos.
   - ➢ También está relacionado con casas_magicas mediante la clave foránea jefe_casa_id en la tabla casas_magicas.

4. cursos_magicos:
   Recoge los datos de los cursos mágicos disponibles, incluyendo duración, nivel, nombre y el profesor que lo imparte.

❖ Relación:
   - ➢ Relación con profesores mediante profesor_id.
   - ➢ Relación con estudiantes a través de la tabla intermedia estudiantes_cursos.

5. criaturas_magicas:
   Recoge información de las criaturas mágicas: ID, descripción, hábitat, nombre y tipo.

❖ Relación:
   - ➢ Está asociada a eventos mágicos mediante la tabla intermedia eventos_criaturas.

6. eventos_magicos:
   Almacena información sobre eventos mágicos: ID, descripción, fecha, nombre y ubicación.

❖ Relación:
   - ➢ Se asocia con estudiantes mediante eventos_estudiantes.
   - ➢ Se conecta con criaturas mágicas mediante eventos_criaturas.

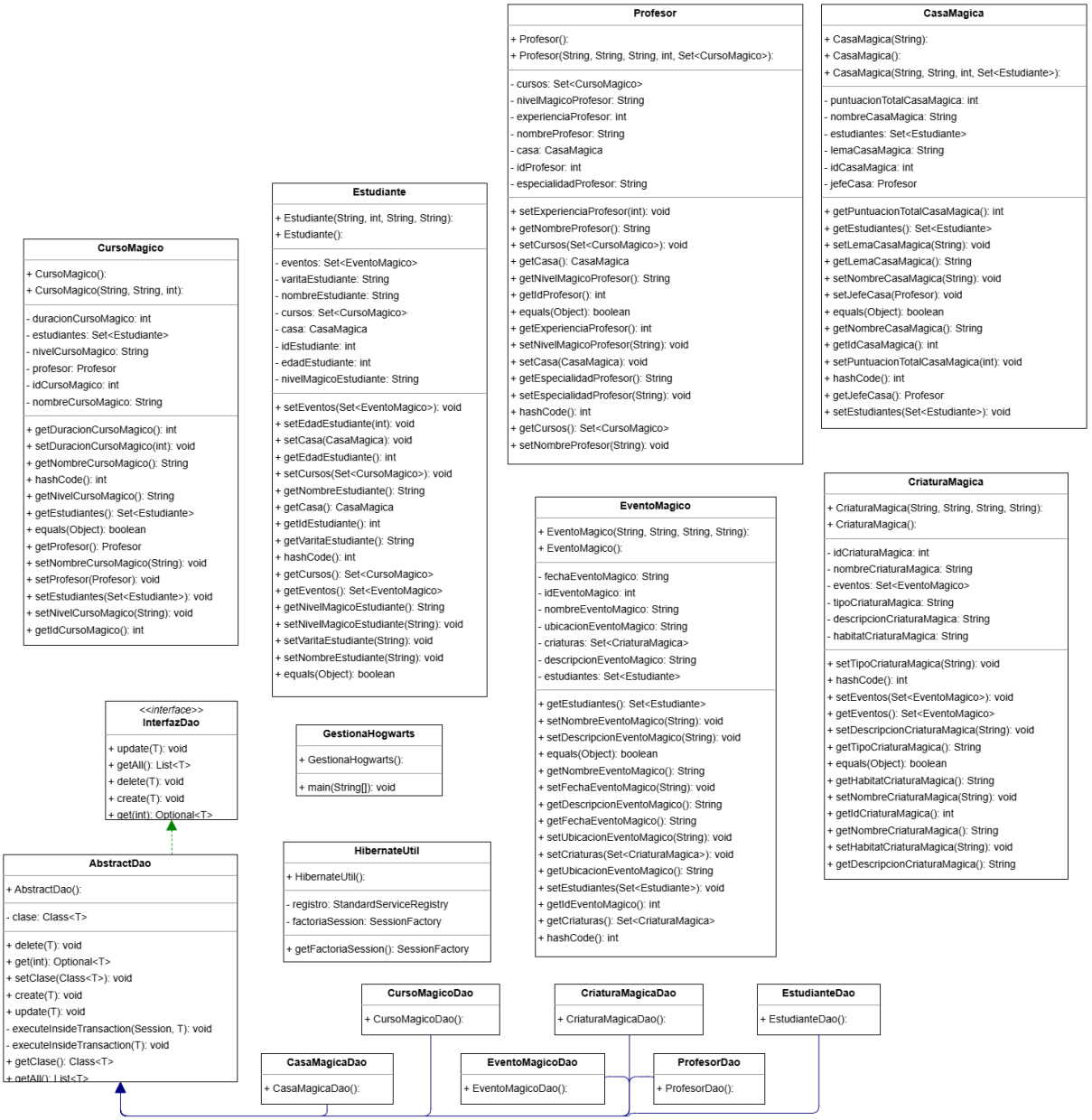
Tablas intermedias (relaciones muchos a muchos)

1. estudiantes_cursos:
   Relaciona estudiantes con los cursos en los que están inscritos.
   Campos: ID de curso e ID de estudiante.
   - Relación muchos a muchos entre estudiantes y cursos_magicos.

2. eventos_estudiantes:
   Conecta a los estudiantes con los eventos mágicos en los que participan.
   Campos: ID de estudiante e ID de evento.
   - Relación muchos a muchos entre estudiantes y eventos_magicos.

3. eventos_criaturas:
   Asocia criaturas mágicas con eventos mágicos en los que aparecen.
   Campos: ID de criatura e ID de evento.
   - Relación muchos a muchos entre criaturas_magicas y eventos_magicos.

4. Define el modelo de clases.



1. **Clases principales:**

- CursoMagico: Representa un curso mágico. Contiene atributos como:
  - idCursoMagico, nombreCursoMagico, duracionCursoMagico.
    - Relaciones con:
      - Un Profesor.
      - Múltiples Estudiante.
- Profesor: Representa a un profesor. Contiene atributos como:
  - idProfesor, nombreProfesor, especialidadProfesor.
    - Relación:
      - Puede impartir varios CursoMagico.
- Estudiante: Representa a un estudiante. Contiene atributos como:
  - idEstudiante, nombreEstudiante, nivelMagicoEstudiante.
    - Relación:
      - Participa en múltiples EventoMagico.
- CasaMagica: Representa una casa mágica. Contiene atributos como:
  - idCasaMagica, nombreCasaMagica, puntajeTotalCasaMagica.
    - Relación:
      - Tiene varios Estudiante.
- CriaturaMagica: Representa una criatura mágica. Contiene atributos como:
  - idCriaturaMagica, nombreCriaturaMagica, tipoCriaturaMagica.
    - Relación:
      - Asociada con múltiples EventoMagico.
- EventoMagico: Representa un evento mágico. Contiene atributos como:
  - idEventoMagico, nombreEventoMagico, fechaEventoMagico.
    - Relaciones:
      - Tiene una lista de Estudiante.
      - Está relacionada con múltiples CriaturaMagica.

2. **Interfaces y clases abstractas:**

- Interfaz InterfazDao<T>: Define operaciones CRUD (Create, Read, Update, Delete) genéricas para la manipulación de datos:
  - create, update, delete, getAll, get.
- Clase abstracta AbstractDao<T>: Implementa la interfaz InterfazDao con métodos concretos que interactúan con Hibernate para realizar operaciones de base de datos.

3. **DAO específicos:**

Son clases que heredan de AbstractDao y se especializan en manejar cada entidad:

- CursoMagicoDao, ProfesorDao, EstudianteDao, CasaMagicaDao, EventoMagicoDao, CriaturaMagicaDao.

4. **Utilidades de Hibernate:**

- HibernateUtil: Proporciona herramientas para la configuración y manejo de sesiones en Hibernate:
  - SessionFactory.
  - Métodos como getSessionFactory() para obtener la sesión de Hibernate.

5. **Clase de gestión:**

- GestionHogwarts: Desde aquí se gestiona el programa.

ENLACE AL REPOSITORIO EN GITHUB