Herramientas de mapeo objeto-relacional (ORM)

Proyecto Unidad 3

Autor:

Antonio Jesús Escudero Godoy

Titulación:

Técnico Superior en Desarrollo de Aplicaciones Multiplataforma

Módulo:

Acceso a Datos

Centro de estudios:

I.E.S. Torre de los Guzmanes

Índice

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

Sprint 1 Demo Day 02/12/24

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.

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.

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

4. Creación de las clases persistentes y mapeo.

5. Crea una clase Servicio que contenga los DAOS y que tenga una operación del CRUD por cada modelo. Estas acciones serán:

1. **Creación y guardado de objetos en BD.**
2. **Borrado de objetos.**
3. **Actualización de objetos.**

6. Crea una clase Controlador desde donde se utilicen cada una de las operaciones definidas.
7. Añade a tus DAOS y a tu clase servicio las siguientes consultas:

7.1. Una consulta a la BBDD que restrinja el número de elementos devueltos de una lista a 1.

7.2. Dos consultas parametrizadas.

Comentarios:

- **No hay que dar las mismas opciones sobre todas las entidades, sino que podéis poner ciertas operaciones básicas sobre algunas de ellas, otras particulares de una entidad concreta, y sobre todo algunas que expliquen el comportamiento de las relaciones entre entidades (por ejemplo: Si borro un departamento, qué ocurre con los empleados que pertenecen a ese departamento y por qué ocurre esto).**
- **Se valorará la variedad e idoneidad de operaciones para explicar cada tipo de relación.**

- **Se tendrá también en cuenta el valor añadido, es decir, que se hayan añadido aspectos que no se pedían como puede ser una interfaz gráfica, o el uso de elementos que no hayamos visto en clase sobre Hibernate,... que hagan que la aplicación sea más profesional.**

ENTREGABLES:

- La entrega se realizará a través de GitHub. Añádeme como colaboradora a tu proyecto (sorayapeceno) y copia la ruta del repositorio. En dicho repositorio debe aparecer:

- Documento con objetivo, Diagrama que represente el modelo relacional de la BD, diagrama de clases, Modelo de datos. Debe estar en la carpeta resources.

- Proyecto con Javadoc generado.

- Script de creación de la BBDD.

Objetivo de tu sistema

El sistema gestiona la información de un cine, incluyendo la organización de salas, la relación entre el cine y su tienda bar, el catálogo de productos, las compras realizadas por los espectadores y las películas visualizadas.

Modelo de datos y sus relaciones

Entidades y sus atributos:

Cine (id, nombre, ubicación)

TiendaBar (id, nombre)

Pedido (id, importe, fecha)

Producto (id, nombre, categoría, precio)

Espectador (dni, nombre, edad)

Película (id, titulo, genero, duración)

Sala (id, nombre, capacidad)

Relaciones:

Relación 1:1

Cine - TiendaBar: Un cine tiene una única tienda bar y una tienda bar pertenece a un solo cine.

Relación 1:N unidireccional

Espectador - Pedido: Un espectador puede o no realizar un pedido y un pedido es realizado por un espectador.

Producto - Pedido: Un producto forma parte de un pedido y un pedido puede estar formado de muchos productos.

TiendaBar - Producto: Una tienda bar tiene varios productos, pero cada producto pertenece a una única tienda bar.

Película - Espectador: Una película puede ser visualizada por varios espectadores y un espectador puede visualiza una película.

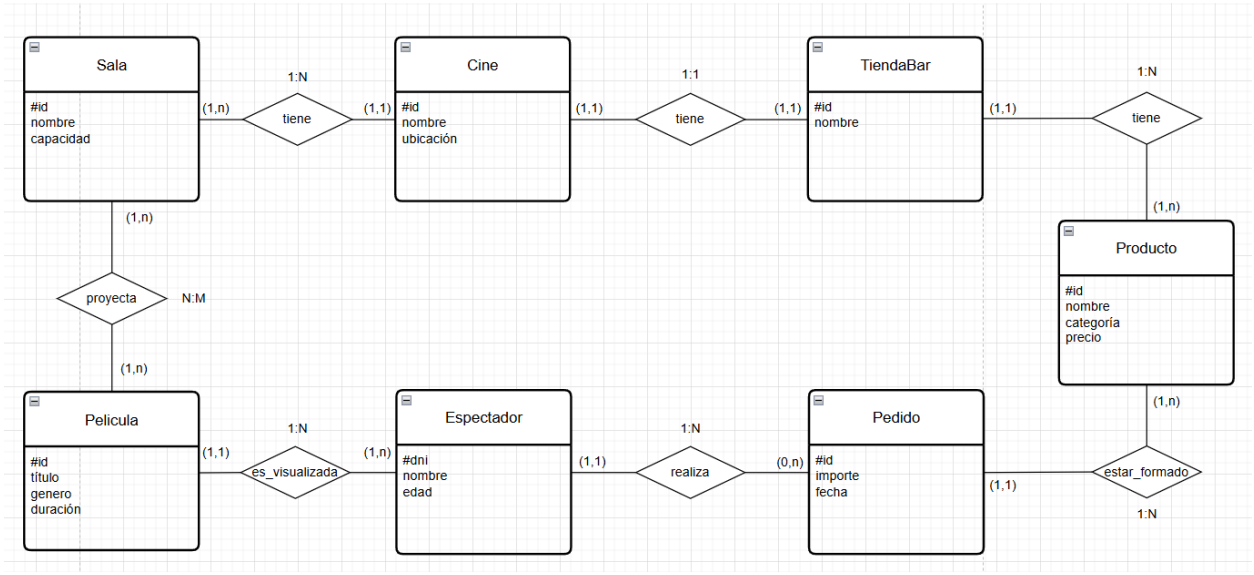
Relación 1:N bidireccional

Cine - Sala: Un cine puede tener varias salas y una sala solo puede estar en un cine.

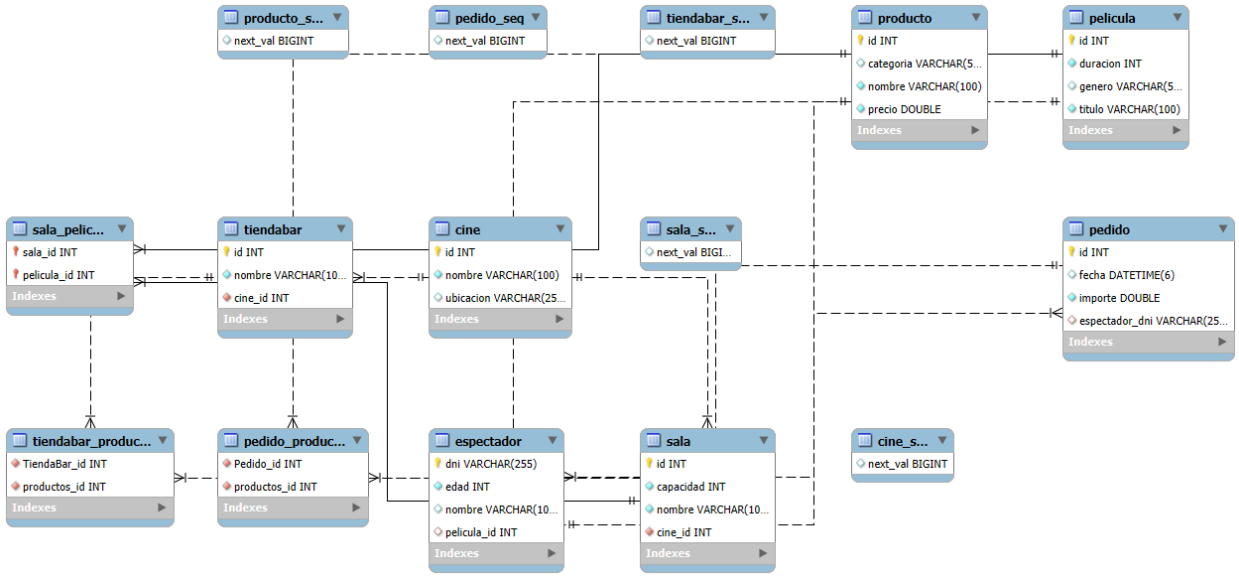
Espectador - Pedido: Un espectador puede o no realizar un pedido y un pedido es realizado por un espectador.

Relación N:M

Sala - Película: Una sala puede proyectar varias películas y una película puede ser proyectada en varias salas.



Esquema del modelo relacional de la BD



Este modelo relacional representa la estructura de una base de datos para la gestión de un cine. Las tablas principales y sus relaciones son:

cine

- Representa un cine.
- Campos:
    - id: Identificador único del cine.
    - nombre: Nombre del cine.
    - ubicación: Ubicación del cine.
- Relaciones:
    - Relación 1:N con la tabla sala (un cine tiene varias salas).
    - Relación 1:1 con la tabla tiendaBar (un cine tiene una tienda bar).

sala

- Representa una sala de un cine.
- Campos:
    - id: Identificador único de la sala.
    - capacidad: Capacidad de la sala.
    - nombre: Nombre de la sala.
    - cine_id: Identificador del cine al que pertenece.
- Relaciones:

    - Relación 1:N con cine.
    - Relación N:M con película a través de sala_pelicula.

pelicula

- Representa una película que se proyecta en el cine.
- Campos:
    - id: Identificador único de la película.
    - título: Título de la película.
    - género: Género de la película.
    - duración: Duración de la película en minutos.
- Relaciones:
    - Relación N:M con sala a través de sala_pelicula.
    - Relación 1:N con espectador.

sala_pelicula

- Tabla intermedia para la relación N:M entre sala y pelicula.
- Campos:
    - sala_id: Identificador de la sala.
    - pelicula_id: Identificador de la película.

espectador

- Representa a los espectadores del cine.
- Campos:
    - dni: Identificador único del espectador.
    - edad: Edad del espectador.
    - nombre: Nombre del espectador.
    - pelicula_id: Identificador de la película que está viendo.
- Relaciones:
    - Relación 1:N con pelicula (un espectador ve una película).
    - Relación 1:N con pedido (un espectador puede hacer varios pedidos en la tiendaBar).

tiendaBar

- Representa una tiendaBar del cine.
- Campos:
    - id: Identificador único de la tienda/bar.
    - nombre: Nombre de la tienda/bar.

    - cine_id: Identificador del cine al que pertenece.
- Relaciones:
    - Relación 1:1 con cine.
    - Relación 1:N con producto a través de tiendabar_producto.

producto

- Representa los productos que se venden en la tienda/bar.
- Campos:
    - id: Identificador único del producto.
    - nombre: Nombre del producto.
    - categoria: Categoría del producto (por ejemplo, bebida, snack, etc.).
    - precio: Precio del producto.
- Relaciones:
    - Relación 1:N con tiendabar a través de tiendabar_producto.
    - Relación 1:N con pedido a través de pedido_producto.

tiendabar_producto

- Tabla intermedia para la relación 1:N entre tiendabar y producto.
- Campos:
    - TiendaBar_id: Identificador de la tienda/bar.
    - productos_id: Identificador del producto.

pedido

- Representa un pedido realizado por un espectador en la tienda/bar.
- Campos:
    - id: Identificador único del pedido.
    - fecha: Fecha y hora del pedido.
    - importe: Importe total del pedido.
    - espectador_dni: Identificador del espectador que realizó el pedido.
- Relaciones:
    - Relación 1:N con espectador.
    - Relación 1:N con producto a través de pedido_producto.

pedido_producto

- Tabla intermedia para la relación 1:N entre pedido y producto.
- Campos:
    - Pedido_id: Identificador del pedido.

- productos_id: Identificador del producto.

Tablas de secuencias

- producto_seq, pedido_seq, tiendabar_seq, cine_seq, y sala_seq son tablas que probablemente gestionan las secuencias automáticas para los identificadores de las tablas principales (producto, pedido, etc.).

Este modelo captura tanto la gestión de películas y salas como la interacción de los espectadores con la tiendaBar del cine.

Script de creación de la BBDD

-- Crear usuario

CREATE USER 'godoy'@'localhost' IDENTIFIED BY '4321';

-- Dar privilegios específicos

GRANT ALL PRIVILEGES ON cinebasedatos.* TO 'godoy'@'localhost';

-- Verificar los privilegios

SHOW GRANTS FOR 'godoy'@'localhost';

-- Eliminar base de datos existente

DROP DATABASE IF EXISTS cinebasedatos;

-- Crear base de datos

CREATE DATABASE cinebasedatos;

-- Usar base de datos

USE cinebasedatos;

Enlace al proyecto en GitHub

https://github.com/ajesgodoy/proyecto_hibernate_ajeg.git