PRÁCTICA 1

BASES DE DATOS

Curso 2021/2022

CREACIÓN Y CONSULTAS A UNA BASE DE DATOS USANDO JAVA

1. Objetivos Generales

- Crear una base de datos en base a un diagrama E-R e insertar el contenido mediante la creación de un programa inicial de carga de datos.
- Acceder a una Base de Datos y realizar consultas SQL mediante un programa implementado en código Java.
- Manejar el conector JDBC (Java DataBase Conectivity), librería de clases para el acceso a un SGBD y para la realización de consultas SQL desde Java.

2. Objetivos específicos

El alumno será capaz de:

- Realizar conexiones a un SGBD desde un programa Java.
- Realizar consultas simples y complejas SQL manejando las clases adecuadas del conector JDBC.
- Tratamiento de los resultados de las consultas SQL dentro del código Java.
- Manejo de las excepciones en la gestión de consultas.
- Implementación de un sistema basado en el concepto de transacción usando código Java.

3. Práctica a realizar

Se desea realizar una aplicación Java que gestione la información de una base de datos que almacena información sobre superhéroes, villanos y películas en las que aparecen. En concreto, se almacena información sobre los superhéroes (con su identificador, nombre y avatar), las personas reales que hay detrás de superhéroes (con identificador, nombre, apellido y ciudad de residencia), los villanos (con identificador y nombre), las películas (con su identificador, título y fecha de estreno), sus escenas (con el número de orden dentro de la película, el título de la escena y su duración).

También se desea conocer qué villanos tienen como rival a un superhéroe concreto y en qué fecha se encontraron por primera vez. De igual forma, es necesario almacenar qué superhéroe protagoniza qué película con qué villano.

La Figura 1 muestra el diagrama E-R que modela la base de datos mencionada.

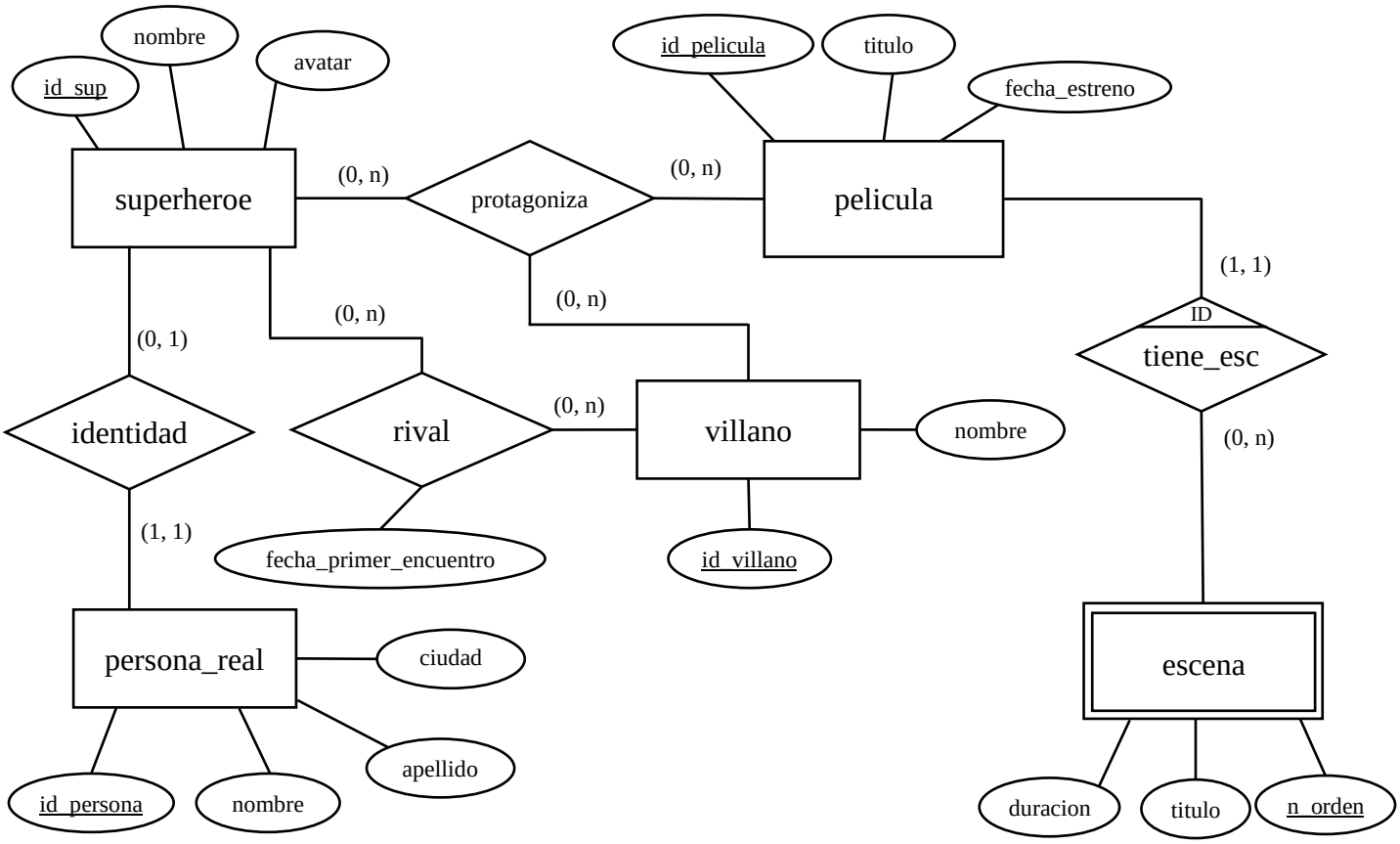

Figura 1. Diagrama entidad relación de la base de datos "Superheroes".

Junto a este enunciado encontrarás cinco elementos:

- *superheroes.sql*: incluye la definición del esquema de la base de datos (y su borrado si ya existe para resetearlo, ¡usar con precaución!), así como la creación de varias tablas y la inserción de datos en ellas. Este script debe importarse o ejecutarse desde el Workbench antes de comenzar a desarrollar la aplicación.
- *superheroes*: paquete de Java que incluye dos clases. La clase *SuperheroesDatabase* contiene el esqueleto de los métodos a implementar y es el único fichero que debe modificarse. El estudiante debe entregar como resultado de esta práctica únicamente el fichero .java que implementa esta clase, completando con el código todo lo que se pide. Por otra parte, el paquete contiene una clase *Main*, que implementa un menú para probar la funcionalidad de la aplicación.

  Importante: queda prohibido modificar la definición de cualquier método ya existente en el fichero. Todos ellos deben hacer un correcto tratamiento de las excepciones. Si se necesita crear algún método adicional, éste debe ser privado.
- *escenas.csv*: fichero CSV que incluye una lista de escenas de varias escenas. Sus filas pueden servir de ejemplo para ser insertadas sin que se produzcan errores (pero en la práctica podrían llegar a producirse) en la tabla *escena* suponiendo que se ha reseteado previamente la base de datos con el script *superheroes.sql*. El orden de los campos está especificado en el ejercicio 3 y cada campo se separa mediante un punto y coma (;). Son los ficheros cargados al llamar a los métodos desde la clase Main.

- *protagonistas.csv*: fichero CSV que incluye una lista de qué superhéroes y qué villanos protagonizan qué películas. Sus filas pueden servir de ejemplo para ser insertadas sin que se produzcan errores (pero en la práctica podrían llegar a producirse) en la tabla *protagoniza* suponiendo que se ha reseteado previamente la base de datos con el script *superheroes.sql*. El orden de los campos está especificado en el ejercicio 4 y cada campo se separa mediante un punto y coma (;). Son los ficheros cargados al llamar a los métodos desde la clase Main.

Con todo ello, se deben implementar los siguientes métodos en Java (además de añadir el código que fuera necesario en cualquier otra parte del fichero):

Primera sesión práctica

1. `openConnection` [0.25 puntos]: este método debe implementar la apertura de la conexión con la base de datos. El método devuelve *true* si abre correctamente la conexión y *false* si ninguna conexión es abierta (bien porque no se pueda o porque ya estuviera abierta). Se debe llamar a este método cuando se vaya a realizar la conexión en otras partes del código. Nunca debe abrir varias conexiones.
2. `closeConnection` [0.25 puntos]: este método debe implementar el cierre de la conexión con la base de datos. El método devuelve *true* si se ejecuta sin errores y *false* si ocurre alguna excepción. No es necesario llamar a este método en el código, ya está llamado donde corresponde en la clase *Main*.
3. `createTableEscena` [0.5 puntos]: este método debe implementar la creación de la tabla *escena*. El método devuelve *false* si la tabla no se ha podido crear (por algún tipo de error o porque ésta ya existía) y *true* si la tabla se ha creado correctamente.

   Prestar atención a cualquier restricción del enunciado. La tabla debe llamarse exactamente *escena*. Se deben nombrar las claves foráneas (si las hubiera) usando exactamente el mismo nombre que la primaria a la que referencian.

   El resto de atributos deben declararse en este orden, llamarse exactamente de la siguiente manera y deben tener los siguientes tipos (si hay claves foráneas, éstas deben ubicarse antes de estos atributos):

   - n_orden → entero
   - titulo → cadena de longitud variable de hasta 100 caracteres
   - duracion → entero
4. `createTableRival` [0.5 puntos]: este método debe implementar la creación de la tabla *rival*. El método devuelve *false* si la tabla no se ha podido crear (por algún tipo de error o porque ésta ya existía) y *true* si la tabla se ha creado correctamente.

   Prestar atención a cualquier restricción del enunciado. La tabla debe llamarse exactamente *rival*. Se deben nombrar las claves foráneas (si las hubiera) usando exactamente el mismo nombre que la primaria a la que referencian. En el orden de los atributos debe aparecer en primer lugar la referencia al superhéroe y después al villano. El atributo con la fecha del primer encuentro entre el superhéroe y el villano debe llamarse exactamente *fecha_primer_encuentro*, debe ser de tipo DATE y su columna será la última de la tabla.

Segunda sesión práctica

5. `loadEscenas` [1 punto]: este método debe insertar en la base de datos las escenas contenidas en el fichero CSV que se pasa como parámetro. El método irá leyendo cada línea del CSV e insertando cada una de ellas (no hay columna de cabecera, pero los atributos aparecen siempre según el orden especificado en el ejercicio 3). Cada inserción debe ser tratada como una transacción separada del resto y en caso de que se produzca un error, el programa debe seguir intentando insertar los elementos posteriores. El método debe retornar la cantidad de elementos insertados en la tabla.
6. `loadProtagoniza` [2 puntos]: este método debe insertar en la base de datos los datos sobre qué superhéroes con qué villanos protagonizan qué películas, que se encuentran en el fichero CSV pasado como parámetro. El método irá leyendo cada línea del CSV e insertando cada una de ellas (no hay columna de cabecera, pero el orden de cada fila siempre es "superhéroe, villano, película"). Cada vez que aparezca una combinación diferente de superhéroe y villano se debe introducir también un registro en la tabla *rival* con ellos. Obligatoriamente, debe ejecutarse la creación y carga de todos los datos como si fuera una única transacción, de tal forma que cualquier fallo intermedio de lugar a deshacer por completo los cambios anteriores. El método debe retornar la cantidad de elementos insertados (la suma de *protagoniza* y *rival*).

Tercera sesión práctica

7. `catalogo` [1 punto]: este método debe consultar en la base de datos la lista de todas las películas y retornar dicha lista como String con la información estructurada de la siguiente forma:

   ```
   {nombre_pelicula_1, nombre_pelicula_2, ..., nombre_pelicula_n}
   ```

   Es decir, debe contener el nombre de todas las películas almacenadas entre llaves y separando cada una de ellas con una coma y un espacio. Además, las películas deben aparecer ordenadas alfabéticamente.

   Si no hay películas almacenadas, debe retornarse "`{}`" (sin las comillas y sin espacios en el interior). Si se produjera alguna excepción, el método debe retornar *null*.
8. `duracionPelicula` [1.5 puntos]: este método debe retornar la duración de una película cuyo nombre se da como parámetro. Dicha duración debe calcularse como la suma de todas las escenas que componen la película. Si no hay ninguna película con ese nombre, debe retornarse -1.0. Si se produce alguna excepción, debe retornarse -2.0. Si la película existe, pero no hay escenas de ella, debe retornarse 0.0.
9. `getEscenas` [2 puntos]: este método debe retornar la lista de nombres de todas las escenas de todas las películas en las que aparezca el villano cuyo nombre se pasa como parámetro con el siguiente formato:

   ```
   {nombre_escena_1, nombre_escena_2, ..., nombre_escena_n}
   ```

   Es decir, debe contener el nombre de todas las escenas almacenadas entre llaves y separando cada una de ellas con una coma y un espacio. Como criterio de ordenación debe usarse, en primer lugar, el orden alfabético de la película y, en segundo lugar, el número de orden de la escena dentro de la película.

En caso de que el villano no exista en la base de datos, o no haya películas para ese villano, o no haya escenas para sus películas, el método debe retornar "`{}`" (sin las comillas y sin espacios en el interior). Si se produjera alguna excepción, el método debe retornar *null*.

Cuarta sesión práctica

10. `desenmascara` [1 puntos]: el método debe obtener la imagen del avatar del superhéroe que enmascara a la persona real cuyo nombre y apellido se pasan en los dos primeros parámetros. La imagen debe almacenarse en la ruta indicada por el parámetro *filename*. Debe retornar *true* si la imagen existía en la base de datos y se ha almacenado correctamente y *false* en caso contrario (en este caso no se debe almacenar nada).

El programa debe conectarse cuando se vaya a realizar una consulta/operación contra la BD por primera vez (no al arrancar el programa) y no debe desconectarse hasta que el programa finalice (de esto último se encarga ya la clase Main).

4. Consideraciones sobre el paquete Java *superheroes*

El paquete de Java *superheroes* contiene los métodos necesarios para ser ejecutado, ya que la clase *Main* contiene el método *main*, que ejecuta el menú del programa. El menú está preparado para que se ejecuten las diferentes operaciones que se pueden escoger (de la 0 a la 6) y que, tras ello, vuelva a mostrarse el menú. Sólo se sale del menú (y del programa) escogiendo la opción 0, que es la de salir. Está prohibido modificar esta clase.

La clase *SuperheroesDatabase* contiene las cabeceras de los métodos que se pide implementar, con lo que el estudiante simplemente debe rellenar el código en dichos métodos. Está prohibido modificar la definición de dichos métodos, pero pueden definirse tantos métodos privados y clases inner nuevas como se deseen. El fichero que contiene esta clase es lo único que se debe entregar.

5. Evaluación y otras indicaciones

Es importante que se cumplan los requisitos establecidos en la práctica, incluyendo:

- El método openConnection() debe ser el encargado de: cargar el driver y realizar la conexión. Es obligatorio que se implemente esta funcionalidad dentro de dicho método. Parámetros:
    - Servidor: localhost:3306
    - Usuario: superheroes_user
    - Password: superheroes_pass
    - Nombre base de datos: superheroes

- Que sean los métodos ya dados los que, al menos, se encarguen de proporcionar el resultado final. El estudiante puede generar otros métodos adicionales si lo considera relevante, pero el método asignado a cada ejercicio debe devolver el resultado.

- El fichero de la clase Java a entregar debe llamarse obligatoriamente "SuperheroesDatabase.java", tal y como se entrega. Es decir, debe usarse en realidad ese fichero, no se debe crear uno nuevo. No debe cambiarse tampoco el nombre del paquete. Solo puede entregarse dicho fichero .java. En caso de necesitar crear clases extra para el procesamiento de datos (aunque no se considera necesario), deben hacerse como clases inner[1].

- La práctica se evaluará de 0 a 10 mediante un corrector automático. Esto quiere decir que los métodos deben funcionar perfectamente y todos los formatos deben ser acordes al presente guion. A la hora de la corrección se usarán tanto los datos especificados aquí, como otros datos para la comprobación de la corrección de las soluciones. Si un método no pasa la prueba automática, se calificará con 0 puntos. Esta corrección automática dará una nota preliminar de la práctica, que puede ser inferior si se dan los supuestos que se enumeran a continuación.
    - **Limpieza y claridad del código** que, además, debe gestionar correctamente los errores, estar optimización, contener comentarios, etc.: no cumplir con estos aspectos puede suponer la deducción de hasta 1.5 puntos.
    - **Fichero entregado con nombre incorrecto o entrega de ficheros adicionales:** -0.5 puntos.
    - **Conexión con un usuario que no sea el dado:** -2 puntos (si es con root), -1 punto (si es con otra combinación de usuario/contraseña inválida).
    - **Realización de más de una conexión/realización de conexión en momento inoportuno:** -1 punto.
    - **No usarse los cierres de las estructuras necesarias:** -0.5 puntos.
    - **No usar PreparedStatement cuando haya parámetro o usarlo incorrectamente (usando concatenaciones):** -3 puntos.
    - **Creación de tablas sin claves foráneas:** -2 puntos.
    - **No identificar correctamente los tipos de errores:** -0.5 puntos.
    - **Lanzamiento de las excepciones hacia arriba:** para esto es necesario modificar las cabeceras de los métodos y supone la calificación automática con un 0.
    - **Usar clases creadas por el grupo y que no sean inner.** El código no funcionará, lo que supone una calificación automática con un 0.
    - Otros supuestos: Los profesores pueden aplicar otras penalizaciones en caso de encontrar errores diferentes a los aquí descritos. Quedará a discreción del profesorado la penalización a aplicar en cada caso.

- La calificación final de la práctica no superará en ningún caso los 10 puntos.

- Los profesores se reservan el derecho de citar a cualquier grupo a defender la práctica si lo considerara necesario.

[1] https://www.tutorialspoint.com/java/java_innerclasses.htm

6. Entrega de la Práctica y resolución de dudas

El grupo de prácticas deberá **entregar mediante la plataforma Moodle en una tarea habilitada para ello un único fichero comprimido (*.zip, *.rar) cuyo nombre sea (Grupo_N_JDBC.rar) (donde N es el número de grupo correspondiente) que contenga:**

1. Clase SuperheroesDatabase.java con la implementación solicitada.
2. Opcionalmente un documento en PDF que contenga comentarios acerca de los problemas surgidos al hacer la práctica o cualquier otro comentario que los alumnos estimen oportuno.

La práctica debe ser entregada por **sólo uno de los miembros del grupo** (preferiblemente el representante).

La fecha límite para la entrega de esta práctica es el viernes **8 de abril de 2022 a las 23:55**.

7. Resolución de dudas

Las dudas deben plantearse en primer a instancia a través del **foro** habilitado para dicho fin en Moodle. Si fuera necesario concertar una tutoría, debe enviarse un correo electrónico al profesor de tu grupo de prácticas.