DAM. UNIDAD 3. ACCESO MEDIANTE MAPEO OBJETO-RELACIONAL (ORM). TAREA EVALUABLE 3

DAM. Acceso a Datos (ADA) (a distancia en inglés)

Unidad 3. ACCESO MEDIANTE MAPEO OBJETO-RELACIONAL (ORM)

Tarea evaluable 3

Abelardo Martínez

Año 2023-2024

Aspectos a tener en cuenta

Importante

Si buscas las soluciones navegando por Internet o preguntando al oráculo de ChatGPT te estarás engañando a ti mismo. Ten en cuenta que ChatGPT no es infalible ni todopoderoso.

Es una gran herramienta para agilizar tu trabajo una vez que dominas un tema, pero utilizarla como atajo a la hora de adquirir habilidades y conocimientos básicos perjudica seriamente tu aprendizaje. Si lo utilizas para obtener soluciones o consejos por tu cuenta, comprueba también cuidadosamente las soluciones propuestas. Intenta resolver las actividades utilizando los recursos que hemos visto y la documentación ampliada que encontrarás en el "Aula Virtual".

Consejos para programar

Aconsejamos seguir las siguientes normas de codificación:

- Una instrucción por línea.
- Añade comentarios para que tu código sea más claro y legible.
- Utiliza la notación húngara para reconocer el tipo de variables a primera vista.
- Recuerde que hay varias formas de aplicar una solución, así que elija la que más le guste. Recomendamos encarecidamente utilizar s o l u c i o n e s basadas en tampones.

A. Instrucciones y directrices

> El proyecto DEBE realizarse en Java. Otras tecnologías -como Spring Boot- no serán soportadas. Para su desarrollo se puede utilizar cualquiera de los IDEs propuestos en la unidad 1, aunque se recomienda encarecidamente Eclipse.

1. VISIÓN GENERAL

Deberá crear por su cuenta una aplicación Java que utilice los conceptos impartidos durante la UNIDAD 3 para cumplir una especificación proporcionada.

2. CALENDARIO Y EXPECTATIVAS

- Porcentajes dentro del TERM: 50% del total del TERM (AT4 haría el otro 50%)
- Porcentajes dentro de la TAREA: 100% competencias ADA (las competencias en inglés deben ser APROBADAS).
- Plazo: 23:59 horas del domingo 7 de enero de 2024 (3 SEMANAS)

3. CALIFICACIÓN

Debes obtener 5 puntos sobre 10 en ADA y un COMPETENTE en inglés para aprobar esta TAREA EVALUABLE.

Junto con este documento se proporcionará una escala de calificaciones detallada (consulte la RÚBRICA DE APRENDIZAJE).

4. RECURSOS

Deberás hacer una lectura exhaustiva de todos los materiales proporcionados por tu profesor, así como de las tareas no evaluables, pero también bucear en Internet para encontrar ejemplos que proporcionen resultados similares a los requeridos por esta tarea.

Siéntete libre de copiar y pegar código de CUALQUIER recurso siempre que entiendas cada parte d e l mismo ya que se te pedirá que defiendas tu trabajo en una reunión individual.

5. PLAGIARISMO

No debes permitir que otros estudiantes copien tu trabajo y debes tener cuidado para evitar que esto ocurra.

En caso de sospecha de plagio, podría exigirse una entrevista oral adicional.

6. ENTREGA Y RETROALIMENTACIÓN

- La tarea se entregará SÓLO en un archivo con formato ZIP, comprimiendo la carpeta del proyecto desde su IDE (es decir, Eclipse).
- Posteriormente, ESTARÁ OBLIGADO a asistir a una entrevista oral con su profesor para discutir ciertos aspectos de su tarea en inglés durante un máximo de 15 minutos.

- Recibirá sus notas desglosadas por cada criterio y el total, junto con cualquier comentario con sugerencias sobre cómo podría haberlo hecho mejor.

B. Detalles de la evaluación

> SÓLO SE PERMITE EL INGLÉS para la realización de la tarea evaluable, tanto los comentarios como los textos explicativos/aclaratorios.

1. CADA MÉTODO DEBE ESTAR DEBIDAMENTE DESCRITO CON SUS PROPIAS PALABRAS. En el
   principio de cada método debes añadir comentarios para explicar con tus propias palabras cómo funciona.
2. ADEMÁS, DEBES AÑADIR UN TEXTO EXPLICANDO CON TUS PROPIAS PALABRAS, TU EXPERIENCIA IMPLEMENTANDO ESTA SOLUCIÓN.

Cree un archivo de texto y cópielo en la carpeta del proyecto o cree el archivo de texto dentro del propio proyecto en el IDE de Eclipse.

- PÁRRAFO 1. Describa brevemente la solución aportada.
- PÁRRAFO 2. Describa brevemente las dificultades encontradas.
- PÁRRAFO 3. Describa brevemente varias posibles ampliaciones que haya recomendado.

B.1. Características obligatorias

Actividad (EVALUABLE)

Cree un programa en Java para gestionar una base de datos de TORNEOS DE AJEDREZ tal y como se muestra en el diagrama Entidad-Relación que puede encontrar más abajo, imprimiendo y utilizando un menú específico. Después de cada opción, el usuario debe ver el mismo menú hasta que se pulse la opción cero. Puede reutilizar el menú que construyó para AT1&AT2. Siéntase libre de duplicar el código y aplicar los cambios necesarios.

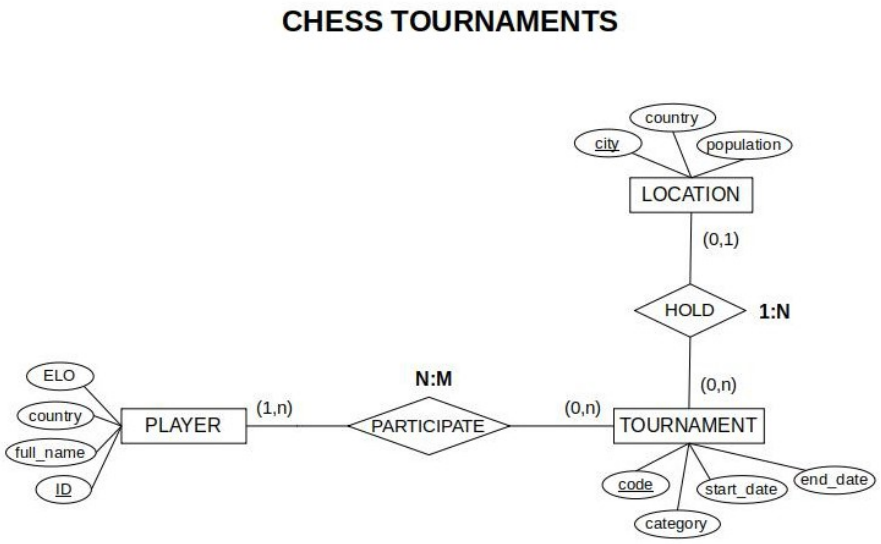
Por favor, siga estas ESPECIFICACIONES TÉCNICAS:

- RDBMS: MySQL
- Idioma: Java
- Framework: Maven
- ORM: Hibernate con anotaciones
- DAO: POJO

ATENCIÓN: Utilice las excepciones adecuadas cuando acceda a bases de datos a través de Hibernate.

---

1) Diagrama entidad-relación y sentencias DDL:



Abelardo Martínez Serrano


```
DROP DATABASE IF EXISTS DBChessTournaments;
CREAR BASE DE DATOS DBChessTournaments CHARACTER SET utf8 COLLATE utf8_spanish_ci

CREATE USER 'mavenuser'@'localhost' IDENTIFIED WITH mysql_native_password BY
GRANT ALL PRIVILEGES ON DBChessTournaments.* to 'mavenuser'@'localhost';

USE DBChessTournaments;

DROP TABLE IF EXISTS Juego;
DROP TABLE IF EXISTS Tournament;
DROP TABLE IF EXISTS Player; DROP
TABLE IF EXISTS Location;

-- Jugador
CREAR TABLA Jugador (
```

```
playerID        INTEGER,
nombre completo
                VARCHAR(100),
país            VARCHAR(50),
ELO             INTEGRO,
CONSTRAINT pla_id_pk PRIMARY KEY (playerID),
CONSTRAINT pla_elo_ck CHECK (ELO > 0)
);

-- Ubicación
CREAR TABLA Localización
( ciudad        VARCHAR(100),
país            VARCHAR(50),
población       INTEGRO,
CONSTRAINT loc_cit_pk PRIMARY KEY (ciudad),
CONSTRAINT loc_pop_ck CHECK (población > 0)
);

-- Torneo
CREAR TABLA Torneo ( código
                VARCHAR(10),
categoría       VARCHAR(20),
fecha_inicio    TIMESTAMP,
fecha_fin       TIMESTAMP,
ciudad          VARCHAR(100),
CONSTRAINT tou_cod_pk PRIMARY KEY
(código), CONSTRAINT tou_cat_ck CHECK   E   ("AFICIONADO", "PROFESIONAL",
(categoría                              N   "MAESTRO",
CONSTRAINT tou_cit_fk FOREIGN KEY (ciudad) REFERENCES Ubicación(ciudad) ON
UPDATE
);

-- Juego
CREAR TABLA Juego (
playerID        INTEGER,
código          VARCHAR(10),
CONSTRAINT gam_pco_pk PRIMARY KEY (playerID, code),
CONSTRAINT FOREIGN KEY (playerID) REFERENCES Player(playerID) ON DELETE CASCA
CONSTRAINT FOREIGN KEY (code) REFERENCES Tournament(code) ON DELETE CASCADE
```

```
);
```

---

2) Opciones de menú:

- Pulse 0 para "Salir".
- Pulse 1 para "Insertar y listar ajedrecistas".
  - Esta opción preguntará por los elementos en bucle hasta que se introduzca cero.
  - Para cada elemento necesitamos ID de jugador (Entero), nombre completo (Cadena con espacios), país (Cadena con espacios) y ELO (Entero > 0).
  - Por cada elemento dado, lo almacenaremos en un ArrayList. Una vez introducido el cero como ID del jugador, todos los elementos se insertarán en la tabla PLAYER y se mostrará una lista de jugadores. Antes de ello, deberías comprobar si el ArrayList está vacío para evitar ejecutar código innecesario.
  - Comprueba si el ID del ajedrecista ya existe en la lista de matrices. En caso afirmativo, debe mostrar un mensaje en pantalla. Debe preguntar por cada valor (en bucle) hasta que el usuario introduzca un ID válido.
  - ATENCIÓN: ¡el ID del jugador y el ELO deben ser un INTEGRO! Para cada JUGADOR, debe preguntar por cada valor (en bucle) hasta que el usuario introduzca un entero válido.
- Pulse 2 para "Insertar y listar torneos".
  - Esta opción preguntará por los elementos en bucle hasta que se introduzca cero.
  - Para cada artículo necesitamos código (cadena sin espacios), categoría (cadena sin espacios), ciudad (cadena con espacios), fecha de inicio (DateTime) y fecha de fin (DateTime).
    - Además, para cada elemento, necesitamos poder asociar tantos ajedrecistas como queramos, hasta que se introduzca cero como ID de jugador (el ID de jugador introducido distinto de cero debe existir).
  - Por cada elemento dado, lo almacenaremos en un ArrayList. Una vez introducido el cero como código, todos los ítems se insertarán en la tabla TORNEO y se mostrará una lista de torneos. Antes de ello, debes comprobar si el ArrayList está vacío para evitar ejecutar código innecesario.
  - Compruebe si el código del torneo ya existe en la lista de matrices. En caso afirmativo, debe mostrar un mensaje en pantalla. Debe preguntar por cada valor (en bucle) hasta que el usuario introduzca un código válido.
  - ATENCIÓN: ¡la fecha de inicio y la fecha de finalización deben ser DateTime! Para cada TORNEO,

debe preguntar por cada valor (en bucle) hasta que el usuario introduzca una fecha válida.

- Pulse 3 para "Insertar y listar ubicaciones".
  - Esta opción preguntará por los elementos en bucle hasta que se introduzca cero.
  - Para cada elemento necesitamos ciudad (cadena con espacios), país (cadena con espacios) y población (entero > 0).
    - Además, para cada elemento, debemos asociar tantos torneos como queramos, hasta que se introduzca cero como código (el código introducido distinto de cero debe existir).
  - Por cada elemento dado, lo almacenaremos en un ArrayList. Una vez introducido el cero como código, todos los ítems se insertarán en la tabla LOCATION y se mostrará una lista de localizaciones. Antes de ello, debes comprobar si el ArrayList está vacío para evitar ejecutar código innecesario.
  - Compruebe si la ciudad ya existe en la lista de matrices. En caso afirmativo, debe mostrar un mensaje en la pantalla. Debe preguntar por cada valor (en bucle) hasta que el usuario introduzca una ciudad válida.
  - ATENCIÓN: ¡la población debe ser un INTEGRO! Para cada UBICACIÓN, debe preguntar por cada valor (en bucle) hasta que el usuario introduzca un entero válido.
- Pulse 4 para "Eliminar ajedrecistas".
  - Esta opción preguntará por los elementos en bucle hasta que se introduzca cero.
  - Para cada ítem dado, lo almacenaremos en un ArrayList. Una vez que se introduzca cero como ID de jugador, se eliminarán (en cascada) todos los elementos de la tabla PLAYER (y tablas asociadas).
- Pulse 5 para "Borrar torneos".
  - Esta opción preguntará por los elementos en bucle hasta que se introduzca cero.
  - Por cada elemento dado, lo almacenaremos en un ArrayList. Una vez que se introduzca cero como código, todos los elementos se eliminarán (en cascada) en la tabla TORNEO (y tablas asociadas).
- Pulse 6 para "Eliminar ubicaciones".
  - Esta opción preguntará por los elementos en bucle hasta que se introduzca cero.
  - Para cada elemento dado, lo almacenaremos en un ArrayList. Una vez introducido el cero como ciudad, todos los elementos se borrarán (en cascada) en la tabla UBICACIÓN y se actualizarán (en cascada) en las tablas asociadas.

Ejemplo de menú:

```
*****
MENÚ
*****
========================================================================
  0. Salida
  1. Insertar y enumerar ajedrecistas
  2. Insertar y listar torneos
  3. Insertar y listar ubicaciones
  4. Eliminar jugadores de ajedrez
  5. Borrar torneos
  6. Eliminar ubicaciones
  7. [opcional] Encontrar un jugador de ajedrez
  8. [Opcional] Buscar un torneo
========================================================================
    Selecciona una opción:
```

B.2. Funciones opcionales

Actividad (EVALUABLE)

Opcionalmente, puede implementar estas siguientes entradas dentro del menú para alcanzar más de 8 puntos sobre 10 en esta TAREA EVALUABLE.

Por favor, siga estas ESPECIFICACIONES TÉCNICAS:

- DQL (Lenguaje de consulta de datos): Criterios HQL

Opciones de menú:

- Pulsa 7 para "[opcional] Encontrar un ajedrecista".
  - Esta opción le pedirá un número (1-9) o un cero para volver al menú.
  - Con ese número, se mostrará una lista de ajedrecistas con ID de jugador que empiece por ese número, ordenados por ID de jugador de forma ascendente.
- Pulse 8 para "[opcional] Buscar un torneo".
  - Esta opción le pedirá una letra (A-Z) o un cero para volver al menú.
  - Con esa letra, se mostrará una lista de torneos cuyo código empiece por esa letra, ordenados por código de forma ascendente.

C. Rúbrica de aprendizaje

C.1. ADA sfiills

Se requiere un mínimo de 5 sobre 10 para esta parte.

Estas notas se invalidarán (nota 4) si no defiende su trabajo en una entrevista oral.

ELEMENTOS DE EVALUACIÓN		DETALLES DEL ELEMENTO DE EVALUACIÓN	PUNTUACIÓN (PUNTOS)
Hibernate/Maven		Hibernate/Maven están configurados correctamente.	0.5
Menú		El menú cumple las especificaciones.	0.5
Objetos POJO		Defines y utiliza las anotaciones correctas de Hibernate. Configura el mapeo ORM correctamente.	2
CRUD operaciones	Insertar datos	Inserta los datos correctamente. Escribe los datos correctamente en MySQL.	2
	Datos de la lista	Lee los datos correctamente desde MySQL. Lista/imprime los datos de forma adecuada.	1.5
	Borrar datos	Borra los datos correctamente de MySQL. Modifica los datos correctamente.	1.5
[opcional] Encontrar un jugador de ajedrez			1
[Opcional] Buscar un torneo			1

C2. Inglés sfiills

Obligatorio ser COMPETENTE para aprobar esta parte.

ELEMENTOS DE EVALUACIÓN	DETALLES DEL ELEMENTO DE EVALUACIÓN	PUNTUACIÓN
Capacidad de redacción	Cada método se describe correctamente. Se proporciona un texto adecuado (dentro del código o en un archivo de texto) para describir la TA utilizando TRES PÁRRAFOS.	COMPETENTE/NO COMPETENTE
Competencias orales	Utiliza un vocabulario adecuado al propósito. Muestra fluencia y confidencia.	COMPETENTE/NO COMPETENTE
Capacidad de comprensión		Cumplido ya que todos los materiales están en inglés
Capacidad de lectura		Cumplido ya que todos los materiales están en inglés