Backend de Aplicaciones 2025

> ## Enunciado de Parcial Turno 2 (Board Games)
>
> **Contexto:** Dataset *Board Games*
>
> **Objetivo:** A partir del pre-enunciado entregado y el DDL `ddl_board_games.sql`, construir una aplicación Java con H2 en memoria que procese e importe los datos provistos en el archivo `board_games_data.csv`, normalizando las entidades y garantizando la integridad de los objetos relacionados. Una vez finalizada la importación, deberán resolverse las consignas del informe detalladas más abajo, aplicando consultas **JPQL** y procesamiento **Java** según corresponda.

Datos

- La estructura de la base de datos y las entidades a mapear se encuentran definidas en el archivo `ddl_board_games.sql` y el pre-enunciado correspondiente.
- El archivo `board_games_data.csv` contiene más columnas que las utilizadas. Solo deberán procesarse aquellas indicadas en el pre-enunciado.
- Los registros deben importarse garantizando la **unicidad** de las entidades relacionadas (`Category`, `Designer`, `Publisher`).
- La ejecución de la aplicación debe realizarse mediante `mvn exec:java`, mostrando resultados por consola en formato legible.

Requerimientos

Nota: Usted dispuso de un pre-enunciado en el que pudo trabajar para preparar las estructuras necesarias para resolver aquí los enunciados que a continuación se solicitan.

Nota: En la solución de estos usted puede utilizar lógica de programación Java, consultas JPQL o una combinación híbrida de ambas herramientas siempre y cuando logre reportar lo pedido al finalizar.

Los requerimientos se dividen en dos secciones:

1. Importación de datos desde un archivo CSV
2. Informe de los resultados solicitados

Importación de datos

Durante la importación del archivo CSV deberán:

1. Procesar los campos relevantes: `name`, `category`, `year_published`, `designer`, `min_age`, `average_rating`, `users_rated`, `min_players`, `max_players`, `publisher`.
2. Si hubiera alguna columna en el CSV que no está relacionada con la base de datos simplemente omitir el valor de esa columna en cada fila (aunque no hay)
3. Para las entidades relacionadas (`Category`, `Designer`, `Publisher`):
    - Verificar si la entidad ya existe en memoria o en la base antes de crearla.

- Puede implementarse una estructura de cache (por ejemplo, `Map<String, Entity>`) para garantizar unicidad.

1 - Informe de resultados

Una vez completada la importación y persistencia, la aplicación deberá resolver las siguientes consultas:

Al finalizar, mostrar por consola un resumen de importación con los siguientes valores:

1. Cantidad de **juegos** importados.
2. Cantidad de **diseñadores**, **categorías** y **publishers** creados.

2 – Análisis global de ratings

> Mostrar las **5 categorías con menor promedio de rating**, considerando únicamente juegos con más de **500 usuarios calificados**.

Indicaciones:

- Realizar la consulta con JPQL (no con SQL nativo).
- Agrupar los resultados por categoría (`Category`).
- Calcular el promedio de `averageRating` y el total de `usersRated` por cada categoría.
- Ordenar por promedio ascendente (de menor a mayor rating).
- Mostrar en consola: `Categoría — Promedio — Usuarios`.

Ejemplo de salida esperada:

```
Categorías con menor promedio de rating (>500 usuarios)
------------------------------------------------------------
Party Games ........... 5.82 (735 usuarios)
Children's Games ...... 5.95 (820 usuarios)
Trivia ................ 6.02 (1200 usuarios)
Abstract .............. 6.10 (610 usuarios)
Card Games ............ 6.14 (980 usuarios)
```

Punto 3 – Juegos disponibles para n jugadores de x edad (con Lógica de dominio)

> Implementar un método que permita determinar si un juego es **apto para una cantidad específica de jugadores** y una **edad dada**, utilizando la información contenida en las columnas `minPlayers`, `maxPlayers` y `minAge`.

Indicaciones:

- Debería evaluar:
  - Si el número de jugadores se encuentra dentro del rango `[minPlayers, maxPlayers]` (considerando rangos abiertos si algún valor es nulo).
  - Si la edad del jugador es mayor o igual al `minAge` requerido.
  - Si el juego tiene al menos 100 ratings

Resultado esperado:

- Listar por consola los juegos que cumplan ambos criterios para un valor recibido por parámetro (por ejemplo, 10 años y 4 jugadores) Ordenados de mayor a menor por raiting.
- Mostrar: `Nombre — Edad mínima — Jugadores (min–max)`.

Ejemplo de salida esperada:

```
Juegos aptos para 4 jugadores y edad 10+
-----------------------------------------
Catan ..................... Edad mínima: 10  Jugadores: 3–4
Ticket to Ride ............ Edad mínima: 8   Jugadores: 2–5
Carcassonne ............... Edad mínima: 7   Jugadores: 2–5
```

Requisitos de la Entrega

Se solicita que el alumno genere una aplicación Java de consola de forma tal que al ejecutar con `mvn exec:java` se ejecute un main que realice las acciones solicitadas en el orden indicado. El main mencionado debe cumplir los siguientes requisitos:

1. Inicializar la base de datos (puede optar por hacerlo de la forma que lo tenga previsto en la realización del pre-enunciado)
    1. La base de datos debe ser implementada con H2 embebido y en memoria
2. Importar el archivo CSV suministrado y cargar la base de datos de acuerdo con la consigna especificada
3. Resolver y mostrar el resultado del punto 1 - Resultados de Importación
4. Resolver y mostrar el resultado del punto 2 - Ranking de peores categorías
5. Resolver y mostrar el resultado del punto 3 para 3 años y 4 jugadores - Juegos para n jugadores de x edad.

Recomendaciones y aclaraciones

- Se brindó el material con Pre-enunciado y Base de datos para que el alumno tuviera una visión previa al día del parcial y que no haya sorpresas, la idea de la cátedra es que durante el parcial codifiquen lo que se pide y entreguen cada uno a su leal saber y entender.
- SOLO tienen que escribir código java y configurar la clase que implemente el método main en el plugin para que el programa se pueda ejecutar con `mvn exec:java` como ya lo hemos realizado en clase en reiteradas oportunidades.
- Para esto:
    - Se espera un trabajo individual y de producción propia, puede utilizar todas las herramientas que tenga a la mano pero no puede consultar con sus compañeros o, menos aún, utilizar el código de estos.
    - Como ya indicamos es su responsabilidad que el proyecto funcione y que el entorno lleve a cabo la tarea para la que se desarrolló

Entrega

Al finalizar la solución debe borrar el dirctorio target con `mvn clean` y luego comprimir el directorio del proyecto y subirlo al aula virtual. Es la intención de la cátedra que las correcciones se lleven a cabo el día del parcial, sin embargo las notas allí vertidas quedarán como tentativas hasta que la cátedra corra los scripts de validación sobre todos los parciales.

La cátedra de Backend les desea todo el Éxito en este Parcial