Examen 3a Eva
Módulo:AADD - Presencial
Apellidos:
Nombre:
Firma:
24-05-2022

Se quiere guardar información de **películas** de **cine**, teniéndose en cuenta que la información a guardar es la siguiente:

- Listado de películas sobre las que guardar información.

- Información en cada **movie** que se debe guardar:
  - **title** – String con el título de la película
  - **duration** – dato numérico de la duración de la película
  - **writer** – guionista/director de la película
  - **year** – dato numérico con el año de estreno
  - **genre** – String con género del film
  - **synopsis** – String con resumen del film
  - **actors** – listado/colección/array… con los actores que participaron en la película.

- Se aportan clases .java con un esqueleto posible para guardar esta información.
  - Por ejemplo, Actor y Writer podrían tratarse como objetos con su propia clase.
  - Por ejemplo, Writer podría tener el listado de movies igual que Actor.
  - No es imprescindible usar estas estructuras

- Se adjuntan también archivos de ejemplo:
  - movies.xml
  - examMongoMovies.js

• Neodatis (3 ptos.)

1- Guardar la información de películas en una Base de Datos **NeoDatis**.

2- Implementar operaciones CRUD desde Java.
- C – inserción de datos para completar info de movies en la BDNeodatis
- R – listado de movies con su info
  - Introduciendo el **nombre** de un **writer**, visualice todas sus **movies**.
  - Introduciendo el **título** de una **movie** visualice los **datos** de la **movie** y el **writer**.
- U – modificar la **duration** de una movie
- D – borrar una determinada movie (conociendo writer y título)

3- Mostrar funcionalidades implementadas con salida desde un **menú**.

- **eXistDB (3 ptos.)**

Partiendo del documento movies.xml, rellenarlo con más datos y realiza consultas en una BD **Exist** con **XQUERY/XPath** desde Java usando:

- XMLDB
- XQJ

4- que muestre los **títles** de **movie** que **empiezen** por **T**.

5-Partiendo del documento anterior realiza una consulta que obtenga el **nombre de writer** de una movie y los **actors** de la misma

6-Visualiza de un **year** determinado el **número** de **movies** que hay.

- **MongoDB (3 ptos.)**

En una base de datos **MongoDB**,

- Desde Java, usando Document, DBCollection, MongoCollection .. (opciones posibles válidas)
- ***Se valorará positivamente resolución utilizando Agregation Framework***

7- Preferentemente usando agregaciones **pipeline** con la colección movies:

- obtener: el título en **mayúsculas**,
- que se visualicen todos los campos menos el id

8-En **MongoDB**, usando agregaciones **pipeline** y la colección movie, obtener:

- por **cada genre**
- la película más reciente (con el **year más alto)**.

9-Usando **MongoDB desde java**, utilizando las etapas de agregado:

- visualiza los **actores**
- de **una película determinada** (utiliza los métodos correspondientes).

10-Usando **MongoDB desde java** y siguiendo con los ejemplos anteriores,

- calcular el **número de películas**,
- **agrupada** por **writer**.

Se valorará: (**1pto**.)

- **Complejidad** de estructura almacenamiento utilizada.
- **Completitud** de las operaciones planteadas.
- Ejecución **efectiva** de los requerimientos.