Práctica 2. Persistencia No Relacional y Evolución.

Enunciado

En esa práctica se parte del enunciado de la práctica anterior, relacionado con una aplicación de gestión de vuelos y mantenimiento de aviones. La descripción de la información a almacenar en la base de datos es la siguiente:

La base de datos gestionará las siguientes entidades, con los datos que se indican para cada una de ellas:

- Avión: matrícula (tipo String), fabricante, modelo, horas de vuelo.
- Aeropuerto: código IATA (aunque son 3 letras se puede utilizar el tipo String), nombre, ciudad, país.
- Tripulante: código de empleado (tipo String), nombre, apellidos, puesto (comandante, co-piloto, sobrecargo, …) y nombre de compañía a la que pertenecen.
- Vuelo: código de vuelo, compañía a la que pertenece, avión que realizó el vuelo, aeropuertos de origen y destino, fecha y hora de salida, duración del vuelo (horas con decimales). Se desea almacenar además los tripulantes de cada vuelo.
- Mecánico responsable de la revisión: código de empleado (tipo String), nombre, apellidos, nombre de la empresa a la que pertenece, año de incorporación a la empresa y formación previa (grado, fp, superior, ...).
- Revisión: avión revisado, fecha de inicio, fecha de fin, número de horas empleadas, mecánico encargado de la revisión, tipo de revisión (periódica, reparación, ...), descripción de trabajos realizados y aeropuerto en que se realizó la revisión.

Se pide realizar la implementación de los apartados siguientes con Java y Spring Data (Java 8) sobre una base de datos MySQL. Para facilitar el desarrollo de la práctica, se puede partir del código entregado previamente en la práctica 1.

Apartado 1: evolución

En este primer apartado se pide la incorporación de Flyway a la aplicación, de manera que tanto la creación de tablas como la inserción de datos se lleven a cabo en la versión 1 de la base de datos. El *DataLoader*, por tanto, no debe insertar datos.

El valor de este apartado en la calificación de la práctica es de un 15 %.

Apartado 2: base de datos híbrida

En este apartado se pide la inclusión de dos atributos JSON así como la migración de datos a esos campos en dos entidades diferentes. Estas transformaciones se harán a través de un script de Flyway, migrando la base de datos a la versión 2. No se deberán borrar los campos ni datos previos.

Dado que la información sobre las tripulaciones es variable porque cada vuelo puede tener un número diferente de tripulantes, se pide crear un nuevo campo JSON que guarde el ID de los tripulantes del vuelo en la entidad que representa la información de vuelos.

También se pide transformar la información de revisiones trasladando esa información a un nuevo campo JSON en la entidad Avion.

Sobre el resultado de la evolución, se pide realizar las consultas siguientes, mostrando su resultado en el *DataLoader*:

- Para cada avión, mostrar el nombre y apellidos de los mecánicos responsables de sus revisiones.
- Para cada tripulante, mostrar su nombre y apellidos junto con su número total de vuelos y la suma de horas de estos.

El valor de este apartado en la calificación de la práctica es de un 60%. Para resolver este apartado pueden resultar útiles las funciones JSON_OBJECT, JSON_ARRAYAGG y JSON_TABLE.

Apartado 3: aggregation framework.

Para trabajar el *aggregation framework* de Mongo se pide incorporar a la práctica una conexión a Mongo, transformando la capa de persistencia de la aplicación en una capa políglota. En este caso, aunque no tenga relación con el escenario anterior, se pide utilizar los datos de ejemplo de provincias disponibles en Aula Virtual (*provincias.json*). En concreto, es imprescindible para la corrección de la práctica que la colección se llame “provincia” y los datos de conexión sean los siguientes:

```
spring.data.mongodb.host=localhost
spring.data.mongodb.port=27017
spring.data.mongodb.database=test
```

Además se pide implementar las siguientes consultas, cuyo resultado se mostrará a través del *DataLogger*:

- Listado de los datos de todas las provincias.
- Listado mostrando, para cada comunidad autónoma, su número de provincias (Ceuta y Melilla se consideran como parte de la comunidad autónoma “sin comunidad”).

Documentación sobre *Aggregation framework* y Spring Data:
https://docs.spring.io/spring-data/mongodb/docs/current/reference/html/#mongodb.repositories.queries.aggregation

Ojo: la anotación @Aggregation admite varios pasos del pipeline:

```
@Aggregation(pipeline = { " { paso1 } " , " {  paso 2 " } )
```

El valor de este apartado en la calificación de la práctica es de un 25 %.

Formato de entrega

La práctica se entregará teniendo en cuenta los siguientes aspectos:

- La práctica se entregará como un fichero .zip que contendrá el proyecto Maven que resuelve la práctica.
  - Solamente hay que incluir *pom.xml* y el directorio *src*.
  - El proyecto se puede crear con cualquier editor o IDE, pero no se deben incluir los ficheros y directorios “de proyecto” del IDE.
- El nombre del fichero .zip será el correo URJC del estudiante (sin @alumnos.urjc.es).
- La práctica se entregará por Aula Virtual según la fecha indicada.

Las prácticas se podrán realizar de forma individual o por parejas. En caso de que la práctica se haga por parejas:

- Sólo será entregada por uno de los alumnos.
- El nombre del fichero .zip contendrá el correo de ambos alumnos separado por guión. Por ejemplo `p.perezf2019-z.gonzalez2019.zip`