Práctica 1. Bases de datos multi-modelo y evolución.

Enunciado

Se desea implementar la capa de persistencia para una aplicación de gestión de asistencia online. En concreto, se trata de almacenar los datos sobre chats que los técnicos y clientes realizan sobre los productos que se venden. En esta implementación se comenzará a trabajar con una versión totalmente relacional que evolucionará a una versión con datos en formato JSON a través de la herramienta Flyway.

La implementación se hará con Java y Spring Data (Java 8) sobre una base de datos MySQL.

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

- Cliente: nombre, apellidos, correo electrónico y ciudad.
- Producto: nombre, marca, precio.
- Técnico: nombre, nivel (dato numérico de 1 a 10).
- Chat: fecha, autor de la entrada (cliente o técnico, puede ser un campo similar a "binario" o *booleano*), texto de la entrada, producto sobre el que se chatea, cliente que lo compró (con quien se está chateando) y técnico que le atiende.

Versión 1: modelo relacional

En la versión 1.0 se deberá tener una base de datos totalmente relacional. Por tanto, se tendrá un script que cree las tablas e inserte los datos iniciales. Se recomienda incluir 3 ó 4 registros en las tablas de clientes, productos y técnicos, y aproximadamente 20 entradas en la tabla de chat. Debería haber algún producto, cliente y técnico que no haya participado en chats.

En la versión 1.0 se deberán implementar las siguientes consultas, cuyo resultado se mostrará debidamente por terminal utilizando una clase similar al *DataLoader* de los ejemplos del curso:

- Listado de chats para un cliente concreto (se puede elegir el que más tenga). El cliente se identificará por su email, no por su *id*.
- Técnicos de nivel menor que 5 que no hayan atendido a ningún cliente.

Tanto en este apartado como en los siguientes, se deben crear consultas con parámetro, de manera que se puedan reutilizar con diferentes valores.

Versión 1.5: datos extra en JSON.

En la versión 1.5 se incorporarán los siguientes elementos:

- Histórico de precios para cada producto. Además del precio actual, se debe añadir un campo tipo JSON a la tabla de productos con un histórico de precios.
  - Ejemplo de contenido del campo: `[{"fecha": "10/10/2018", "precio": 15.50}, {"fecha": "10/10/2018", "precio": 18.50}]`
- Listado de habilidades de cada técnico. Se deberá añadir un listado de etiquetas que indiquen las distintas habilidades del técnico.
  - Ejemplo de contenido del campo: `["Móviles","Domótica"]`

Además de modificar adecuadamente las entidades, se deberá crear un fichero para migración desde la versión 1.0 que deberá tener los correspondientes comandos *ALTER TABLE*, así como un conjunto de sentencias para inserción de datos de prueba.

En esta versión se deberán implementar las siguientes consultas:

- Técnicos con la habilidad “Wearables”.
- Productos que alguna vez hayan tenido un precio menor que 10 euros.

Versión 2.0: chats en JSON

La versión final de la base de datos transformará la tabla de chat en una tabla con clave principal más un campo JSON con toda la información agrupada por cliente, producto y técnico. Un ejemplo de fila de la tabla sería la siguiente:

```
{"tecnico": 4, "cliente": 2, "producto": 9, "chat": [ {"fecha": "01/02/2019 10:50", "autor": "C", "texto": "Hola, tengo problemas con el portátil. No se enciende."}, {"fecha": "01/02/2019 10:52", "autor": "T", "texto": "Hola, le atiende Paco. ¿Ha probado a enchufarlo a la corriente?." } ] }
```

Se pide implementar las consultas de la versión 1 más las siguientes:

- Listado de todos los chats de una fecha dada. Elegir una con más de una línea de chat.
- Listado de chats sobre el producto de nombre “Home Pod”.

Además, se deberán incluir sentencias que, a través de Java (no del Script), hagan los siguientes cambios:

- Añadir un valor al histórico de precios de un producto.
- Borrar la habilidad “Video” de todos los técnicos con nivel mayor que 3.
- Insertar una línea de conversación de chat para uno de los casos.

Prueba de la práctica

Es necesario que la práctica funcione correctamente para los tres escenarios. Estos se probarán con la base de datos vacía ejecutando secuencialmente los tres escenarios desde cero. Es decir, arrancando y parando la aplicación cada vez. Se deberá diseñar un mecanismo para ejecutar las consultas solamente si las tablas están preparadas. Es decir, no se pueden mostrar las consultas de la versión 1.5 si la base de datos sigue en la versión 1. Como opción básica se permite el caso simple de manualmente indicar en el código la versión en que está la base de datos. Se valorarán otras propuestas.

Los datos a utilizar en la práctica deben permitir ilustrar todos y cada uno de los ejemplos que se piden.

Formato de entrega

La práctica se entregará teniendo en cuenta los siguientes aspectos:

- La práctica se entregará como un fichero .zip del proyecto Maven. El nombre del fichero .zip será el correo URJC del alumno (sin @alumnos.urjc.es).
    - Solamente hay que incluir *pom.xml* y directorio *src*.
- El proyecto se puede crear con cualquier editor o IDE.
- 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`