DigitalHouse>

Certified Tech Developer
The Ultimate Degree

Back End I

CLASE 21 - TEST 02

1. **Hibernate** mapea las clases Java en tablas de BD y provee mecanismos para consultar datos. El mapeo lo realiza a través de:
   - Configuración XML
   - Anotaciones
   - Configuración XML o Anotaciones

   Hibernate *permite mapear clases Java a tablas de base de datos utilizando tanto* ***configuración XML*** *como* ***anotaciones****. Los desarrolladores pueden elegir cualquiera de estos enfoques para definir el mapeo de objetos a tablas, o incluso combinarlos si es necesario. Las* ***anotaciones*** *se han vuelto más comunes porque integran el mapeo directamente en las clases de entidad.*

2. **Hibernate** es un framework ORM que ayuda a lograr la persistencia de datos.
   - Verdadero
   - Falso

   Hibernate *es un framework* ***ORM (Object-Relational Mapping)*** *que facilita la persistencia de datos al mapear las clases Java a tablas en una base de datos relacional. Proporciona una capa de abstracción sobre la base de datos, lo que permite interactuar con los datos utilizando objetos Java en lugar de escribir directamente sentencias SQL, haciendo más sencilla la gestión de la persistencia de datos en aplicaciones Java.*

3. Cuando anotamos una clase con **@Entity** y no especificamos el nombre de la tabla con la anotación **@Table** en su atributo "name", entonces nos dará un error de mapeo de datos.
   - Verdadero.
   - Falso.

   Cuando anotamos una clase con ***@Entity*** *y no especificamos el nombre de la tabla con la anotación* ***@Table(name = "nombre_de_tabla")****, Hibernate utiliza el nombre de la clase como el nombre de la tabla de forma predeterminada. Por lo tanto, no dará un error de mapeo de datos a menos que haya alguna otra discrepancia o conflicto. La anotación* ***@Table*** *es opcional si deseas usar un nombre de tabla diferente al de la clase.*

4. Cuando queremos etiquetar a la clase como un Bean que va a ser mapeado por el ORM con una tabla de la BD, utilizamos la anotación:

   - @Entity
   - @Table
   - @Column

   *La anotación **@Entity** se utiliza para marcar una clase Java como un **Bean** que será mapeado por el ORM (como Hibernate) a una tabla en la base de datos.*
   *Esta anotación indica que la clase es una entidad persistente y que sus instancias se deben guardar en la base de datos. La anotación **@Table** se usa para especificar el nombre de la tabla (opcional), y **@Column** se usa para mapear los atributos de la clase a las columnas de la tabla.*

5. Si la forma de gestionar los IDs de la tabla que estamos mapeando se realiza a través de una secuencia creada automáticamente por Hibernate, decimos que la anotación **@GeneratedValue** —en su atributo "strategy = GenerationType"— tiene el valor:

   - IDENTITY
   - AUTO
   - TABLE
   - SEQUENCE

   Cuando Hibernate utiliza una secuencia para gestionar los IDs de una tabla, se debe usar la anotación `@GeneratedValue` con la estrategia `GenerationType.SEQUENCE`. Esto le indica a Hibernate que los valores de los identificadores deben generarse usando una secuencia de base de datos.

6. Cuando definimos una interfaz en la capa repository (o DAO), al extender de JpaRepository, solo podemos hacer uso de los métodos delete y update.

   - Verdadero
   - Falso

   Al extender de `JpaRepository`, se tiene acceso a una amplia variedad de métodos, no solo a los métodos `delete` y `update`. Esta interfaz también incluye otros métodos como `save`, `findById`, `findAll`, `deleteById`, entre otros, que permiten realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) de forma completa y fácil sobre las entidades gestionadas por JPA.