HW#4

1 – What is JPA ?

Java Persistence API is the technology that enables associating Java classes with relational database tables and to persist the data after the application terminates. The purpose is to match class variables with the columns of the table and to perform database operations directly on objects without writing SQL. At this point, JPA only sets a standard and does not take any action on the data itself.

Therefore, there is a need for a separate tool that implements these standards, *Hibernate, TopLink, EclipseLink and OpenJPA* can be counted as the main tools that implement JPA.

2 - What is the naming convention for finder methods in the Spring data repository interface ?

Spring Data JPA repository provides us with some predefined method to perform the basic create, read, update and delete (CRUD) operations. Spring data JPA has its own naming conventions for methods.

The derived method mainly has *two* parts separated by ***"By"*** to indicate the start of the actual criteria:

1. The first part is the **find** → what you want to perform with the method
2. And second is ***By*LastName** (for example) → where (field name) you want to perform the operation

3 - What is PagingAndSortingRepository ?

PagingAndSortingRepository is an extension of CrudRepository to provide additional methods to retrieve entities using the pagination and sorting abstraction. It offers two methods :

- Page findAll(Pageable pageable) – returns a Page of entities meeting the paging restriction provided in the Pageable object.
- Iterable findAll(Sort sort) – returns all entities sorted by the given options. No paging is applied here.

4 - Differentiate between findById() and getOne() ?

In Spring Data JPA, both getOne() method and findById() method can be used to fetch an object.

- The findById() method is available in CrudRepository interface, getOne() method is available in JpaRepository interface.
- Calling findById() returns an eagerly fetched entity. Calling getOne() returns a lazily fetched entity.
- findById() method return the actual objects and the entity fields contain the value from the database. getOne() returns a reference of the entity. All fields can contain default values.
- Calling getOne() when the object needs to be accessed, it throws EntityNotFoundException if the object is not in the database. Calling findById() returns null if the object is not registered in the database.

The main difference between these two methods is related to performance, getOne() with Lazy Load is better in performance than findById() as it does not go from JVM to database.

5 - What is @Query used for ?

If the commands we want to write are too complex for the JPA class to understand, the SQL query is declared to the relevant command with the @Query annotation and the specified query is executed when the method is called.

```
@Repository
public interface BookRepository extends CrudRepository<Book, Integer> {
@Query("SELECT b FROM Book b")
List<Book> findAllBooks();
}
```

6 - What is lazy loading in hibernate ?

Lazy loading is a fetching approach used for all the entities in Hibernate. It decides whether to load a child class object while loading the parent class object. When we use association mapping in Hibernate, it is required to define the fetching technique. The main purpose of lazy loading is to fetch the needed objects from the database.

Let's say we have user entity and address entity associated with User. When I request information about the User table, I do not want the address information to come. In this case, the name of the structure we use is lazy loading. In other words, when the actual object is called, the associated object does not come.

```
@OneToMany(fetch = FetchType.LAZY)
```

7 – What is SQL injection attack ? Is Hibernate open to SQL injection attack ?

SQL injection is a malicious code injection technique that can destroy your database.With SQL injection, attackers on the website can steal user information, access hidden information, interfere with existing data, change some operations, increase their authority, delete the database completely.
The most common risk in SQL injection attacks is that email addresses and logins are stolen and sold on the dark web. For this reason, a successful SQL injection poses a great threat not only to institutions, but also to users and customers.
Hibernate does not guarantee immunity to SQL Injection,the api can be misued.

8 - What is criteria API in hibernate ?

Criteria is a very powerful API. It is very simple to use and is used to define queries for entities and their persistent state by creating query-defining objects.

The Criteria API allows you to build up a criteria query object programmatically; the *org.hibernate.Criteria* interface defines the available methods for one of these objects. The *Hibernate Session* interface contains several overloaded `createCriteria()` methods.
Simple example of a criteria query with no optional parameters or restrictions;

```
Criteria crit = session.createCriteria(Product.class);
List<Product> results = crit.list();
```

This query will return every object that corresponds to the class.

The Criteria API makes it easy to use restrictions in your queries to selectively retrieve objects;
Simple example of a criteria query with "equals" restriction;

```
Criteria crit = session.createCriteria(Product.class);
crit.add(Restrictions.eq("description","Mouse"));
List<Product> results = crit.list()
```

This query will will search all products having description as "Mouse".

9 - What Is Erlang? Why Is It Required For Rabbitmq ?

Erlang is a functional, general-purpose language oriented towards building scalable, concurrent systems with high availability guarantees.
The main thing that distinguishes Erlang from other languages is its process-based computing model. It uses isolated, lightweight processes that communicate with each other through messages. These processes can receive messages and, in response to messages, create new processes, send messages to other processes, or modify their state.
The RabbitMQ server program is written in the Erlang programming language and is built on the Open Telecom Platform framework for clustering and failover.

10 – What is the JPQL ?

JPQL (Java Persistence Query Language) is a platform-independent query language defined by the JPA standard for querying entity objects. Its structure and syntax are very similar to SQL. However, JPQL uses the entity object model instead of database tables to define a query. That makes it very comfortable to write queries.
JPQL provides two methods that can be used to access database records.

*1. We can use **createQuery()** method of **EntityManager** interface to create an instance of Query interface for executing JPQL statement.*

```
Query query = entitymanager.createQuery("Select UPPER(e.ename) from Employee e");
List<String> list = query.getResultList();
```

*2. We can use **@NamedQuery** annotation.*
The createNamedQuery() method of EntityManager interface is used to create an instance of Query interface for executing named queries.
We use this annotation in our entity class. It has 2 attributes which is name and query.

```
@NamedQuery(query = "Select c from Customer c where c.id = :id", name = "find customer by id")
```

Then we can execute this query in our main class.

```
Query query = entitymanager.createNamedQuery("find customer by id");
```

11 – What are the steps to persist an entity object ?

1) *Creating an entity manager factory object*
   `EntityManagerFactory emf=Persistence.createEntityManagerFactory("Student_details");`

2) *Obtaining an entity manager from factory.*
   `EntityManager em=emf.createEntityManager();`

3) *Initializing an entity manager.*
   `em.getTransaction().begin();`

4) *Persisting a data into relational database.*
   `em.persist(s1);`

5) *Closing the transaction*
   `em.getTransaction().commit();`

6) *Releasing the factory resources.*
   `emf.close();`

12 – What are the different types of entity mapping ?

- **One To One Relationship:** One entity bean relates only to one other entity bean.
  For example Each student must complete one worksheet (and the worksheet can only be completed by one student).
- **One to Many or Many to One Relationship:** it's a relationship that links one entity to many other entities.
  For example; Let's say we have a teacher and a course entity.
  A teacher can give multiple courses(*One-to-Many - one teacher to many courses*), but a course is given by only one teacher(*Many-to-One - many courses to one teacher*)
- **Many To Many Relationship:** In a many-to-many relationship, many objects can reference many objects. Effectively, in a database, a *Many-to-Many* relationship involves a middle table referencing both other tables.
  Let's say we have a student and a course entity. A student can attend multiple courses, and a course can be followed by multiple students. (*Many-to-Many* relationship)

13 - What are the properties of an entity ?

The data values associated with an entity consist of one or more properties. Each property has a name and one or more values. A property can have values of more than one type, and two entities can have values of different types for the same property.

14 - Difference between CrudRepository and JpaRepository in Spring Data JPA?

JpaRepository extends CrudRepository and it inherits some of the methods available in the CrudRepository like findOne, get and so on. And JpaRepository also extends PagingAndSorting Repository uses for Pagination and Sorting purpose.

CrudRepository	JpaRepository
1. CrudRepository extends Repository interface.	1. JpaRepository extends PagingAndSortingRepository and QueryByExampleExecutor interface.
2. CrudRepository provides methods to perform CRUD operations.	2. JpaRepository provides additional methods like flush(), saveAndFlush(), deleteInBatch() etc.
3. The saveAll(Iterable<S> entities) method of CrudRepository returns Iterable.	3. The saveAll(Iterable<S> entities) method of JpaRepository returns List.
4. If we have to perform mainly CRUD operation, define our repository using CrudRepository.	4. If we have to perform CRUD as well as Batch operation define our repository extending JpaRepository.

Aleyna ZENGİN