Spring Data Jpa

As soon as we add spring data jpa we have to provide properties of datasource (its replacement for spring orm, spring hibernate, spring transaction), it is compulsory.

Now we can configure our datasource and hibernate as

By providing the following in application.properties

spring.datasource.url=
spring.datasource.username=
spring.datasource.password=
spring.jpa.properties.hibernate.dialect=

For entity it also expects a default constructor.

Spring Data JPA and Spring Hibernate are two different approaches for working with databases in a Spring application:

1. **Spring Data JPA**: Spring Data JPA is part of the larger Spring Data project, which aims to simplify database access in Spring applications. Spring Data JPA provides a higher-level abstraction over JPA (Java Persistence API), making it easier to work with JPA entities and repositories. It provides repository interfaces that you can extend to perform CRUD operations on entities without writing explicit data access code.

2. **Spring Hibernate**: Hibernate is a popular object-relational mapping (ORM) framework for Java applications. Spring Hibernate integration allows you to use Hibernate as the ORM solution in your Spring applications. With Spring Hibernate integration, you can configure Hibernate session factory beans, transaction management, and other Hibernate-related components using Spring's configuration mechanisms.

Here are some key differences between Spring Data JPA and Spring Hibernate:

- **Abstraction Level**: Spring Data JPA provides a higher-level abstraction compared to Spring Hibernate integration. It simplifies data access by providing repository interfaces and leveraging JPA annotations for entity mapping. Spring Hibernate integration requires more configuration and manual setup of Hibernate-specific components such as session factories and transaction managers.

- **Ease of Use**: Spring Data JPA simplifies data access by providing repository interfaces with predefined methods for common CRUD operations. Spring Hibernate integration requires you to write more boilerplate code for configuring Hibernate components and performing data access operations.

- **Supported ORM Frameworks**: Spring Data JPA is specifically designed for JPA-based ORM frameworks such as Hibernate, EclipseLink, etc. It provides support for multiple JPA implementations. Spring Hibernate integration focuses specifically on integrating Hibernate with Spring applications.

In summary, Spring Data JPA provides a higher-level abstraction for working with JPA entities and repositories, while Spring Hibernate integration offers more flexibility and control over Hibernate-specific configuration and features. The choice between them depends on your application requirements, familiarity with JPA and Hibernate, and preferred level of abstraction.

Query DSL -> query domain specific language.
JPA provides the implementation of some methods depending on the properties or variables following convention (like method starting by findBy/ getBy will find by some properties of the domain like it should be findByStudentName if the name parameter in student class is studentName, likewise we can query for findByStudentNameAndStudentId).

We can also order by like findByAnameOrderById(String aname); or findByAnameOrderByIdDesc(String aname);

We can write also our own query:

@Query annotation:

@Query(“from Alien where aname = :name”)
List<Alien> find(@Param(“name”) String name);

Here :name is placeholder for name and we have to clarify that using @Param.