**Difference between JPA, Hibernate and Spring Data JPA**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Aspect | JPA (Java Persistence API) | Hibernate | Spring Data JPA |
| **1** | Type | Specification (API) | Implementation (Framework) | Spring module |
| **2** | Provider | Java EE / Jakarta EE | Red Hat | Spring Team (VMware) |
| **3** | Purpose | Defines standard for ORM | Provides ORM and extra features | Simplifies ORM using JPA + Spring |
| **4** | Abstraction Level | High-level interface | Mid-level (requires configs) | Higher abstraction (minimal code) |
| **5** | Need for Implementation | Yes, needs one like Hibernate | No, already an implementation | Uses JPA provider internally |
| **6** | Ease of Use | Requires boilerplate code | Easier with annotations | Easiest with repository abstraction |
| **7** | Configuration | Manual setup required | Needs XML or annotation config | Auto-configured with Spring Boot |
| **8** | Default Implementation | None | Hibernate (commonly used) | Hibernate or any JPA provider |
| **9** | Query Language | JPQL | HQL (Hibernate Query Language) | JPQL, HQL, Native SQL, Method Names |
| **10** | Native SQL Support | Yes | Yes | Yes |
| **11** | Caching | Not defined in spec | Yes (First & Second Level) | Supports if Hibernate is used underneath |
| **12** | Transactions | Supports via @Transactional (external config) | Manages with JTA or JDBC | Integrated with Spring’s @Transactional |
| **13** | Lazy Loading | Defined in spec | Fully supported | Inherited from JPA/Hibernate |
| **14** | Entity Mapping | Uses annotations like @Entity, @Id | Same, plus additional annotations | Same annotations used |
| **15** | Repository Support | Not available | Not built-in | Provides CrudRepository, JpaRepository |
| **16** | Method Query Derivation | No | No | Yes, via method naming conventions |
| **17** | Paging and Sorting | Manual implementation | Manual implementation | Built-in via PagingAndSortingRepository |
| **18** | Auditing (Created/Updated Dates) | Not included | Requires manual handling | Supported with annotations like @CreatedDate |
| **19** | Boilerplate Reduction | Low | Moderate | High – drastically reduces code |
| **20** | Best Use Case | When using plain Java EE | Custom implementations or advanced ORM | Spring Boot applications with minimal code |