**JPA, HIBERNATE AND DATA JPA**

All these are related to ORM

**Object-Relational Mapping (ORM)** is a technique that bridges the gap between object-oriented programming languages and relational databases. It allows developers to interact with database data using familiar programming language objects instead of writing raw SQL queries.

**JPA (Java Persistence API): A Specification/Standard**

* JPA is a *specification* (a set of interfaces and annotations) that defines how to manage relational data in Java applications. It's not an implementation itself, but rather a standard API for ORM.
* It provides a standardized way for Java developers to interact with databases using Java objects, abstracting away the underlying SQL. This means you can write your persistence logic using JPA, and then choose any JPA *provider* (implementation of JPA) to execute it.

**Hibernate: A JPA Implementation/ORM Framework**

* Hibernate is a popular, open-source *implementation* of the JPA specification. It's an ORM framework that provides the concrete code to actually perform the object-relational mapping defined by JPA.
* Hibernate handles the translation between Java objects and database tables, generating SQL queries behind the scenes.

**Spring Data JPA: An Abstraction/Simplification Layer**

* Spring Data JPA is an *abstraction layer* built on top of JPA. It aims to simplify the development of data access layers in Spring applications significantly.
* **Purpose:** To reduce boilerplate code and increase developer productivity by providing a powerful repository abstraction. It minimizes the need to write repetitive CRUD (Create, Read, Update, Delete) methods and even custom queries.

