**JPA:**

Conjunto de reglas y normas y especificaciones de cómo hacer el mapeo y la persistencia de aplicaciones java a bases de datos relacionales.

Lo implementan los frameworks como: [ORM :: Object Relational Mapping]

Hibernate, EclipseLink, TopLink.

Que hacen: convierten los objetos java hacia bases de datos relacionales.

Clase [Obj] = > tabla.

Atributo => columna.

**ORM:**

**mapeo de objeto-relacional**, es un modelo de programación que consiste en la transformación de las tablas de una base de datos, en una serie de entidades que simplifiquen las tareas básicas de acceso a los datos para el programador.

**Por qué usarlo:**

Aunque el lenguaje SQL se usa para acceder a muchas de las bases de datos existentes, existen múltiples varianzas en las funciones que los distintos SGBD han usado. Entonces si cambiamos de gestor tendríamos que cambiar también la sintaxis en cada consulta donde esta se vea afectada, sin embargo, con la ORM no tendríamos ese problema ya que se enlaza de manera general y este se realice automáticamente.

De la misma manera si aumentaríamos o restaríamos columnas a las tablas SOLO tendríamos que alterar en la entidad principal y listo, ya no en cada consulta SQL que ejecutemos en la app.

Esto el ORM al tener una capa intermedia, **abstrae al programador de la base de datos y le centra en el desarrollo de la aplicación**.

Otro punto importante es la **facilidad de trabajo**, un ORM, nos facilita las labores básicas de cualquier acceso a datos, el CRUD (Create, Read, Update y Delete). Realizando todas estas labores a través de un lenguaje de alto nivel orientado a objetos.

* Ventajas
  + Facilidad y velocidad de uso
  + Abstracción de la base de datos usada.
  + Seguridad de la capa de acceso a datos contra ataques.
* Desventajas
  + En entornos con gran carga poner una capa más en el proceso puede mermar el rendimiento.
  + Aprender el nuevo lenguaje del ORM.