# 01\_Introducción a los ORM

* Las siglas ORM significan “Object-Relational mapping” [1)](http://cursohibernate.es/doku.php?id=unidades:01_introduccion_orm:06_intro_orm" \l "fn__1) y en castellano es “Mapeo Objeto-Relacional”. El ORM es simplemente el código que escribimos para guardar el valor de nuestras clases en una base de datos relacional.
* Es decir, es un framework de persistencia de nuestros datos a una base de datos.
* Es decir que Hibernate es simplemente una serie de clases Java [4)](http://cursohibernate.es/doku.php?id=unidades:01_introduccion_orm:06_intro_orm" \l "fn__4) que nos ayudaran a insertar, leer, actualizar borrar nuestros objetos en la base de datos.

## Historia de los ORM en Java (Tecnologías de persistencia)









* Realiza sólo una cosa y la hace bien: Sólo se puede persistir a bases de datos relacionales, a diferencia de JDO que permite otros tipo de repositorios.
* ¿Pero cuál es, entonces, la desventaja de Hibernate? La principal desventaja es que al no haber una especificación detrás de él , si usas Hibernate ya no puedes cambiar de framework de persistencia tan fácilmente como podrías usando JDO o EJB.



Mi opinión personal es que no entiendo para qué crear JPA existiendo ya JDO.

JPA es **el estándar** de persistencia en Java y para aupar más aún a JPA ,Hibernate implementa la especificación de JPA pero no la de JDO. Así que ahora podemos usar Hibernate siguiendo un estándar por lo que podríamos cambiar de Hibernate a otra implementación sin problemas.

## HIBERNATE

Para hacer la persistencia de nuestros objetos Java a la base de datos hay que indicar cómo se debe realizar dicha persistencia. Para ello hay dos métodos:

* Ficheros XML
* Anotaciones en el código.

Las anotaciones que tiene Hibernate son actualmente las del estándar de JPA. De esa forma al ver las anotaciones de hibernate estaremos viendo las de JPA.Cuando veamos los ficheros XML usaremos los ficheros propietarios de Hibernate, así también veremos cómo funciona realmente Hibernate.