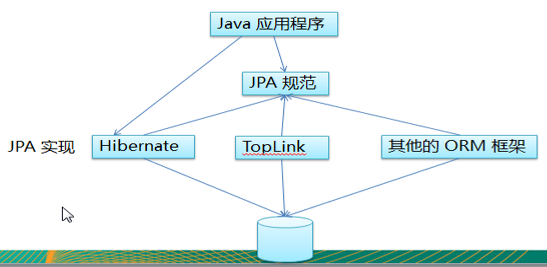
# 佟刚JAP知识讲解：

## 一、JPA概述？

### 1、JPA是什么？

1）JPA，即 Java Persistence API，用于对象持久化的API。

2）Java EE 5.0 平台标准的ORM，是的应用程序以统一的方式访问持久层。



### JPA与Hibernate(或其它ORM框架)的关系

1. JPA是规范(标准)，而不是框架：

JPA本质上就是一种ORM规范，不是ORM框架。因为JPA并为提供ORM实现，它只是制定了一些规范，提供了一些编程的API接口（没有实现类），但具体实现由ORM厂商提供实现。

1. Hibernate是实现：

Hibernate除了作为ORM框架之外，它也是一种JPA实现。

3）从功能上来说，JPA是Hibernate功能的一个子集。

### 3、JPA的供应商：

1）JPA的目标之一是制定一个可以由很多供应商实现的API，目前HIbernate 3.2+、TopLink 10.1+、以及OpenJPA都提供了JPA的实现

2）Hibernate：

【1】JPA的始作俑者就是Hibernate的作者

【2】Hibernate从3.2开始兼容JPA

1. OpenJPA：

OpenJPA是Apache组织提供的开源项目

1. TopLink：

### 4、JPA的优势：

1）标准化：提供同一套API，这保证了基于JPA开发的企业应用能经过少量的修改就能在不同的JPA框架下运行

2）简单易用，集成方便：JPA的主要目标之一就是提供更加简单的编程模型，在JPA框架下创建实体和创建Java类一样简单，只需要使用 java.persistence.Entity 进行注解。

3）可媲美JDBC的查询能力：JPA的查询语言是面向对象的，JPA定义了独特的JPQL，而且能够支持批量更新和修改、Join、groupby、having等通常只有SQL才能提供的高级查询特性，设置还能够支持子查询。

4）支持面向对象的高级特性：JPA中能够支持面向对象的高级特性，如类之间的继承，多态和类之间的复杂关系，最大限度的使用面向对象的模型。

### 5、JPA包括3方面的技术：

1）ORM映射元数据：JPA支持XML和JDK 5.0 注解两种元数据的形式，元数据描述对象和表之间的映射关系，框架据此将实体持久化到数据表中。

2）JPA的API：用来操作实体对象，执行CURD操作，框架在后台完成所有的事情，开发者从繁琐的JDBC和SQL代码中解脱出来。

3）查询语言（JPQL）：这是持久化操作中很重要的一个方面，通过面向对象而非面向数据库的查询语言查询数据，避免数据和具体的SQL紧密耦合。

### 6、JPA 规范要求在类路径的META-INF目录下放置persistence.xml文件，文件的名称是固定的。

## 二、JPA入门HelloWorld：

### 1、使用JPA持久化对象的步骤：

1）创建 persistence.xml ,在这个文件中配置持久化单元：

【1】需要指定跟哪个数据库进行交互

【2】需要指定JPA使用哪个持久化的框架以及配置该框架的基本属性

1. 创建实体类，使用annotation 来描述实体类跟数据库表之间的映射关系
2. 使用JPA API完成数据CRUD操作。

【1】使用EntityManagerFactory（对应Hibernate的SessionFactory）

【2】创建EntityManager（对应Hibernate的Session）

### 2、参考案例【JPADemo1/test1】

## 三、JPA注解：[参考JPADemo1]

### 1、@Entity：

1）@Entity注解用于在实体类声明语句之前，指出该java类为实体类，将映射到指定的数据库表。如声明一个实体类 Customer，它将映射到数据库中的 customer 表。

### 2、@Table：

1）当实体类与其映射的数据库表名不同名时，需要使用 @Table 注解说明数据库表的名字。

2）@Table注解的常用属性是name，用于指明数据库表的名字

3）catalog和schema属性通常用于设置表所属的数据库目录或模式，通常为数据库名

4）uniqueConstraints属性用于设置约束条件，通常不需设置

### 3、@Id：

1）@Id注解用于声明一个实体类的属性映射为数据表的主键列。该注解通常置于属性声明语句之前，可与声明语句同行，也可写在单独行上。

2）@Id注解也可置于属性的getter方法之前

### 4、@GeneratedValue：

1）@GeneratedValue注解用于标注主键的生成策略，通过strategy属性指定。默认情况下，JPA自动生成一个最适合底层数据库的主键生成策略：SQLServer对应identity；MySQL对应auto。

2）在 java.persistence.GenerationType 中定义了以下几种可供选择的策略：

【1】IDENTITY：采用数据库id自增长的方式来自增主键字段，Oracle不支持这种方式。

【2】AUTO：JPA自动选择合适的策略，是默认选项

【3】SEQUENCE ：通过序列产生主键，通过@SequenceGenerator注解指定序列名，MySQL不支持这种方式。

【4】TABLE：通过表产生主键，框架借由表模拟序列产生主键，使用该策略可以使应用更易于数据库移植。

①将当前主键的值单独保存在一个数据表中，主键的值每次都是从指定的表中查询来获得

②这种方法生成主键的策略可以使用于任何数据库，不必担心不同数据库不兼容造成的问题。

【参考JPADemo1/test3】

### 5、@Basic：

1）@Basic注解表示一个简单的属性到数据库表的字段的映射，对于没有任何标注的getter方法，默认即为@Basic

2）fetch：表示该属性的读取策略，有EAGER和LAZY两种，分别表示“主支抓取”和“延时加载”，默认为EAGER

3）optional：表示该属性是否允许null，默认为true

### 6、@Column：

1）当实体的属性与其映射的数据表的列名不同名时需要使用@Column标注说明。

2）@Column注解的常用属性是name，用于设置映射数据表的列名。此外，该注解还包含其他属性，如：unique、nullable、length等

3）@Column标注的columnDefinition属性：表示该字段在数据库中的实际类型。通常ORM框架可以根据属性类型自动判断数据库中字段的类型，但是对于Date类型任然无法确定数据库中字段类型究竟是DATE、TIME、还是TIMESTAP。此外，String的默认类型为varchar，如果要将String类型映射到特定数据库的BOLB或TEXT字段类型

### 7、@Transient：

1）@Transient注解标注的属性（或getter方法），表示该属性并非一个到数据库表的字段的映射，ORM框架将忽略该属性。

2）如果一个属性并非数据库表的字段的映射，就务必将其标注为@Transient，否则ORM框架默认为其加上@Basic注解

### 8、@Temporal:

1）在核心Java API中并没有定义Date类型的精度。而在数据库中，表示Date类型的数据有DATE、TIME、TIMESTAMP，即日期、时间、两者兼备。在进行属性映射时可使用@Temporal注解来调整精度。

【参考JPADemo1/test2】

## 四、JPA API：

### 1、JPA先关接口/类：Persistence

1）Persistence类用于获取EntityManagerFactory实例，该类包含有一个名为 createEntityManagerFactory() 的静态方法

2）createEntityManagerFactory()方法有如下两个重载版本：

【1】createEntityManagerFactory(String persistenceUnitName)：以JPA配置文件persistence.xml中持久化单元名为参数

【2】createEntityManagerFactory(String persistenceUnitName, Map properties)：第一个参数是persistence.xml中持久化单元名称；第二个参数是Map类型，用于设置JPA的相关属性，这时将忽略其他地方设置的属性。Map对象的属性名必须是JPA实现库提供商的名字空间约定的属性名。

即：像下面圈出来的之类的名字

|  |
| --- |
|  |

【参考：JPADemo2/testPersistence】

### 2、EntityManagerFactory：（类似Hibernate的SessionFactory）

1）EntityManagerFactory接口主要用来创建EntityManager实例。该接口中定义了如下4个方法

【1】createEntityManager()：用来创建“实体管理器”对象实例

【2】createEntityManager(Map map)：用于创建“实体管理器”对象实例的重载方法，Map参数用于提供EntityManager的属性

【3】isOpen()：检查EntityManagerFactory是否处于打开状态。“实体管理器工厂”创建后一直处于打开状态，除非调用close()方法将其关闭

【4】close()：关闭EntityManagerFactory。EntityMangerFactory关闭后将释放所有资源，isOpen()方法测试返回false，其他方法将不能调用。

### 3、EntityManager：（类似Hibernate的Session）

【注： 持久化上下文也就是一级缓存 ，参考网址：http://yuquan.blog.51cto.com/1924539/360703/】

1）find()：查询，类似于Hibernate的get()方法

2）getRefernce()：查询，类似于Hibernate的load()方法

3）persist()：保存，类似于Hibernate的save()方法

4）remove()：删除，类似于Hibernate的delete()方法

【参考JPADemo2/testEntityManager/TestEntityManager.java】

5）merge()：用来处理Entity的同步，即数据库的插入和更新操作

【1】实体的状态：

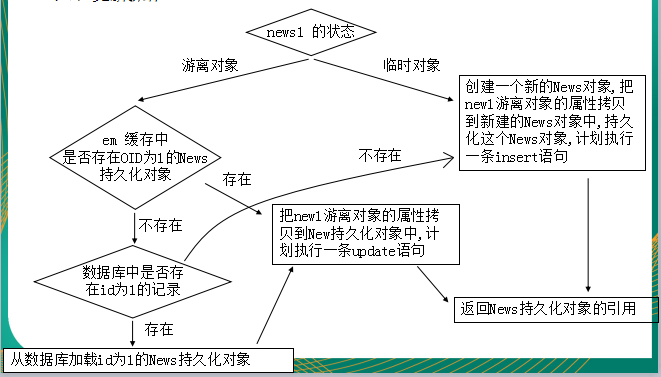
①新建状态: 新创建的对象，尚未拥有持久性主键。

②持久化状态：已经拥有持久性主键并和持久化建立了上下文环境

③游离状态：拥有持久化主键，但是没有与持久化建立上下文环境

④删除状态: 拥有持久化主键，已经和持久化建立上下文环境，但是从数据库中删除。

【2】图解说明：



【参考JPADemo2/testEntityManager/TestMergeOfEntityManager.java】

1. flush()：同步持久上下文环境，即将持久上下文环境的所有未保存实体的状态信息保存到数据库中 。
2. setFlushMode(FlushModeType flushMode)：设置持久上下文环境的Flush模式，参数可以取两个枚举值：

【1】FlushModeType.AUTO：自动更新数据库实体

【2】FlushModeType.COMMIT：提交事务时才更新数据库记录

8）getFlushMode()：获取持久上下文环境的Flush模式。返回FlushModeType类的枚举值。

9）refresh(Object entity)：用数据库实体记录的值更新实体对象的状态，即更新实体的属性值。

10）clear()：清楚持久上下文环境，断开所有关联的实体，如果这是还有为提交的更新则会被撤销。

11）contains(Object entity)：判断一个实体是否属于当前持久上下文环境管理的实体

12）isOpen()：判断当前的实体管理器是否是打开状态

13）getTransaction()：返回资源曾的事务对象。EntityTransaction实例可以用于开始和提交多个事务

14）close()：关闭实体管理器，之后若调用实体管理器实例的方法或其派生的查询对象的方法都将抛出 IllegalstateException 异常，除了getTransaction 和 isOpen方法(返回 false)。不过，当与实体管理器关联的事务处于活动状态时，调用 close 方法后持久上下文将仍处于被管理状态，直到事务完成。

15）createQuery(String qlString)：创建一个查询对象

16）createNamedQuery(String name)：根据命名的查询语句块创建查询对象。参数为命名的查询语句

17）createNativeQuery (String sqlString)：使用标准 SQL语句创建查询对象。参数为标准SQL语句字符串。

18）createNativeQuery (String sqls, String resultSetMapping)：使用标准SQL语句创建查询对象，并指定返回结果集 Map的 名称。

### 4、EntityTransaction：

1）EntityTransaction接口用来管理资源层实体管理器的事务操作。通过调用实体管理器的getTransaction()方法获得其实例。

## 五、映射关联关系：

### 1、映射单向多对一的关联关系：

【参考JPADemo3/manyToOneOfOneway】

### 2、映射单向一对多的关联关系：

【参考JPADemo2/oneToManyOfOneway】

### 3、映射双向一对多（多对一）的关联关系：

【参考JPADemo2/oneToManyOfTwoway】

### 4、映射双向一对一的关联关系

## 六、JPQL：

### 1、JPQL概述：

1）JPQL语言，即 java Persistence Query Language的简称。JPQL是一种和SQL非常相似的中间件和对象化查询语言，它最终会被编译成针对不同底层数据库的SQL查询，从而屏蔽不同数据库的差异。

2）JPQL语言的语句可以是select语句、update语句或delete语句，他们都是通过Query接口封装执行。

### 2、Query接口：

1）Query接口封装了执行数据库查询的相关方法，调用EntityManager的createQuery()、createNamedQuery()、createNativeQuery()方法可以获得查询对象，进而可调用Query接口的相关方法来执行查询操作。

2）Query接口的主要方法：

【1】executeUpdate()：用于执行update或delete语句

【2】getResultList()：用于执行select语句并返回结果集实体列表

【3】getSingleResult()：用于执行之返回单个结果实体的select语句

【4】setFirstResult()：用于设置从哪个实体记录开始返回查询结果

【5】setMaxResult()：用于设置返回结果实体的最大数。与setFirstResult()结合使用和实现分页查询。

【6】setFlushMode()：设置查询对象的Flush模式，参数可以去2个枚举值：FlushModeType.AUTO，为自动更新数据库记录；FlushModeType.COMMIT，为直到提交事务才更新数据库记录。

【7】setHint(String hintName,Object Value)：设置与查询对象相关的特定供应商参数或提示信息，参数名及其取值需要参考特定JPA实现库提供商的文档，

【8】setParameter(int position,Object value)：为查询语句的指定位置参数赋值。position：指定参数序号；value：赋给参数的值

【9】setParameter(int position,Date date,TemporalType type)：为查询语句的指定位置赋Date值。