**一．Hibernate-BASE**

**hibernate操纵数据库从本质上而言就是调用session的各种API函数来实现的。**

**ORM**: Object/Relationship Mapping 对象/关系映射

**背景**：利用面向对象思想编写的数据库应用程序最终都是把对象信息保存在关系型数据库中，于是要编写大量和底层数据库相关的SQL语句，这种方法存在很多问题：

1. 不同的数据库使用不同的SQL语句，ORACLE=PL/SQL Microsoft=T/SQL
2. 同样的功能在不同的数据库中有不同的实现方式

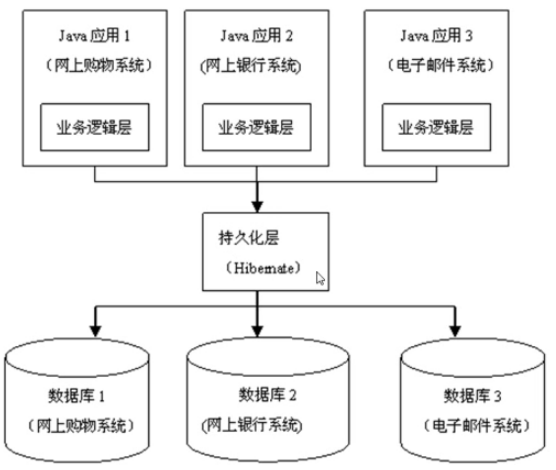
比如分页SQL; ORACLE=ROWNUM MySQL=LIMIT SQL SERVER=TOP

3.程序过分的依赖SQL对程序的移植和扩展，维护带来很大的麻烦

**作用**：ORM框架技术，彻底抛弃写SQL的思想，完全的使用面向对象的思想开发软件

**Hibernate：**Java领域一款开源的ORM框架技术，对JDBC进行了轻量级的封装【本质上还是通过JDBC对数据库的操作】，充当持久化层的角色

**持久化层的作用**：把程序中生成的对象持久化到数据库当中【白话：把生成的对象通过Hibernate或对象关系映射最终保存在数据库的表中】



**主流的ORM框架技术：**

1. MyBatis：前身就是著名的iBatis Apache开发
2. Toplink：被ORACLE收购，重新包装为ORACLE AS TOPLINK
3. EJB：本身是JavaEE的规范，重量级的ORM框架技术，比较复杂应用较少

**Hibernate开发环境：**eclipse+Hibernate Tools for Eclipse Plugins+jar

**Hibernate Tools** :是由JBoss推出的一个Eclipse综合开发工具插件，该插件可以简化ORM框架Hibernate,以及JBoss Seam,EJB3等的开发工作

**Hibernate Tools：**

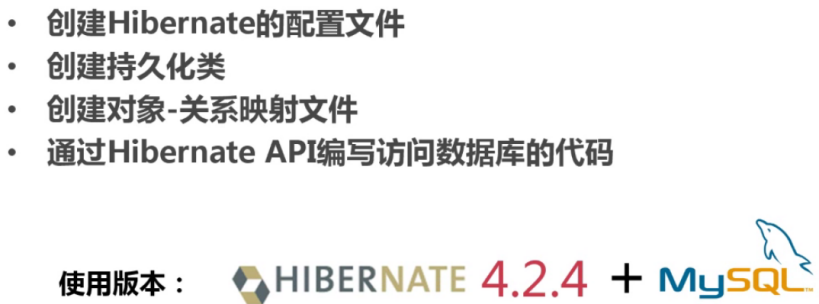
安装过程：Eclipse Help----Install New Software----Add----Archive[.rar压缩文件，Name随便起]----√Next----Finish----YES

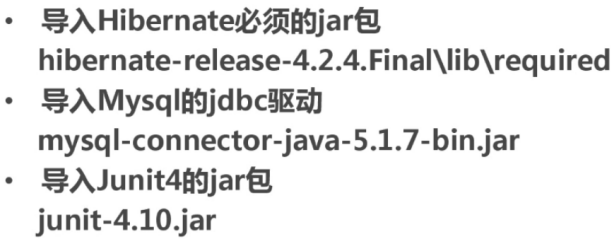
**或**：Eclipse Help----Install New Software---【<http://download.jboss.org/jbosstools/mars/stable/updates/>】注意eclipse和java版本号

测试：File----New----Other----Hibernate【存在代表安装成功】

**Hibernate FIrst DEMO:**

**基本步骤概要：**





--------------------------------------------------------------------------------------------

**开始**：

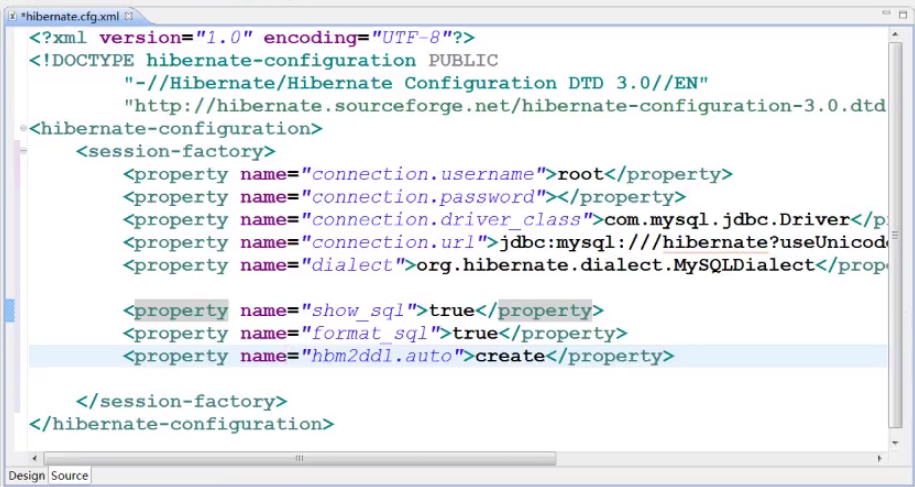
**1**.创建hibernate工程----一个普通的Java Project

**2**.创建hibernate配置文件---src---new---Other---hibernate---hibernate.cfg.xml

【通过tools插件自动生成，提示:如果生成的xml配置文件编辑时没有自动代码提示功能，需要在eclipse中导入dtd文件的支持，该dtd文件路径】







**3.**创建持久化类：遵循JavaBeans的设计原则，例如Students.java

**4.**创建对象-关系映射文件src---new---Other---Hibernate XML Mapping file(hbm.xml)----选中写好的Students.java

**5**.返回hibernate.cfg.xml---添加<mapping resource=”Students.hbm.xml”/>映射

**6**.创建数据库,编码utf-8

**7**.使用JUnit进行单元测试，通过hibernate API编写访问数据库的代码，详见HibernateBase

三个注解：@Test:测试方法 @Before:初始化方法 @After:释放资源

New---Source Folder---test---创建测试类Test---包含三个方法

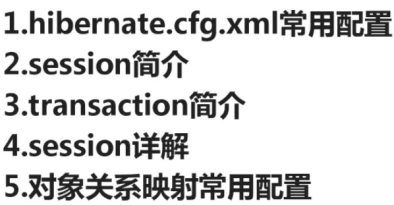
@Before public void init(){}

@Test public void test(){}

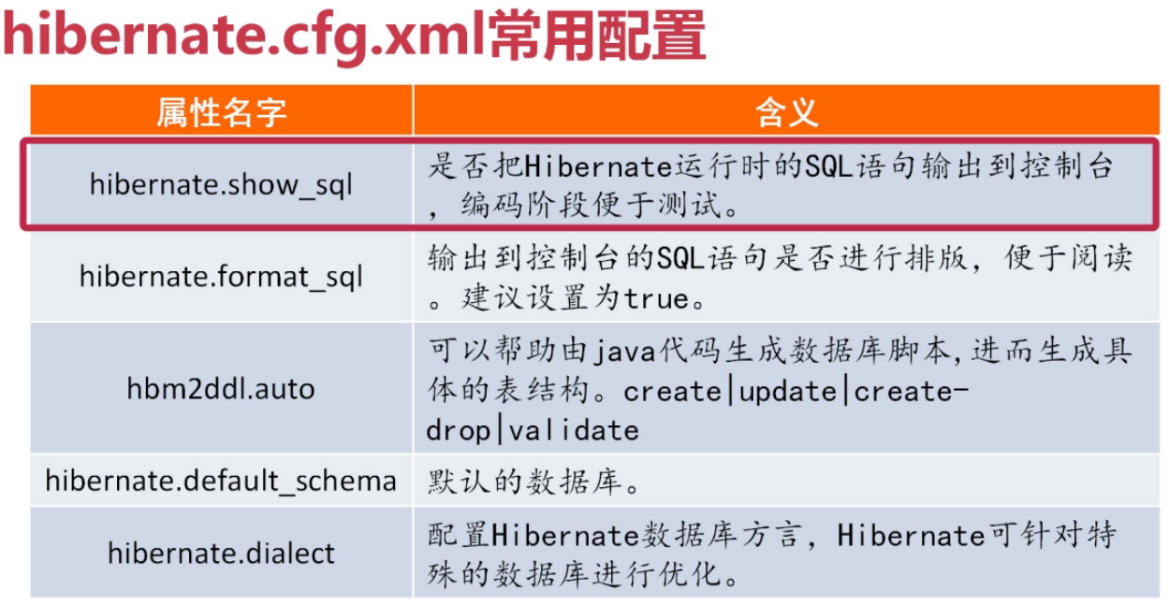
@After public void destory(){}

**二．Hibernate-Advanced**

**内容概要：**



**1.hibernate.cfg.xml常用配置**



**所有属性名的前缀hibernate可以省略**

**Hibernate.default\_schema:**在创建表时会默认的给表名之前加上前缀

**Hbm2ddl.auto：**

Create创建新的表结构，若已存在同名的表，会先删除再创建并添加记录

Update 在原有的表基础上进行更新

Create-drop 先创建再删除

Validate 对原有的表结构进行验证，如果现有的表结构和原有的表结构不同就不会创建表

**---------------------------------------------------------------------------------------------------------**

1. **session简介**

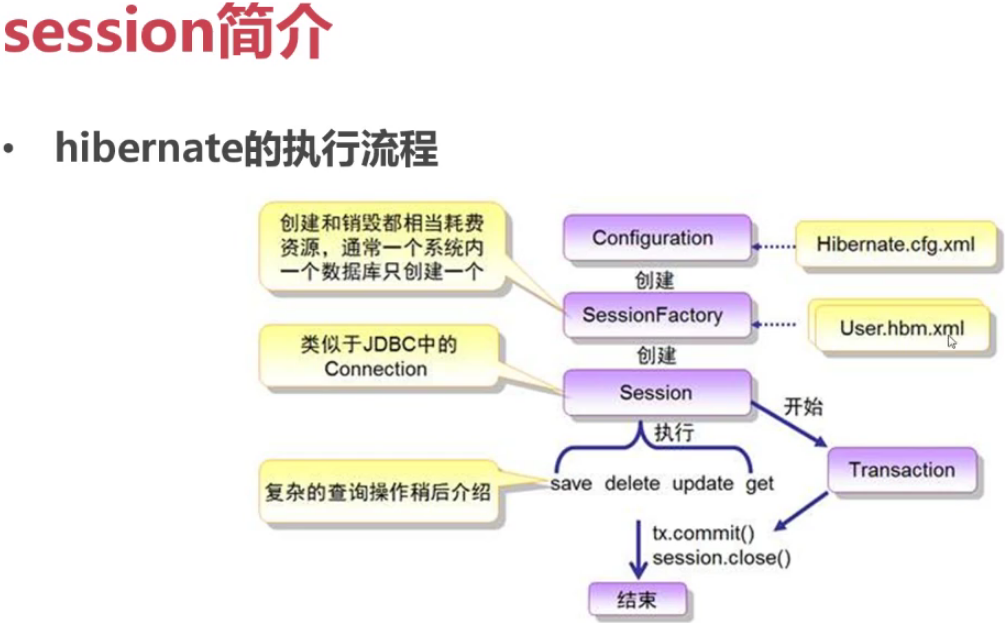
Hibernate是对JDBC的一个封装，它不建议直接使用JDBC的connection操作数据库，而是通过session操作数据库。

所以session就是一个操作数据库的对象，在使用hibernate操作数据库之前，就必须要先获得一个session的实例，session对象表示数据库的一个连接。

Session和JDBC的connection可以理解为是多对一的关系，每个session都有一个与之对应的connection，一个connection不同时刻可以供多个session使用

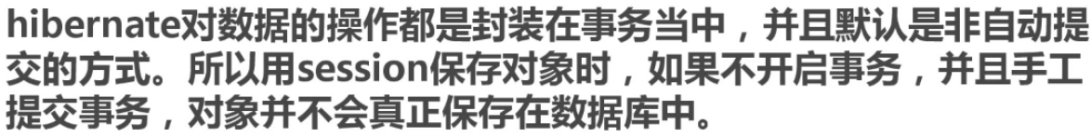
要想把对象保存在关系型数据库中

**需要调用session的各种方法，如save() update() delete() createQuery()等**



**--------------------------------------------------------------------------------------------**

**3.transaction简介**

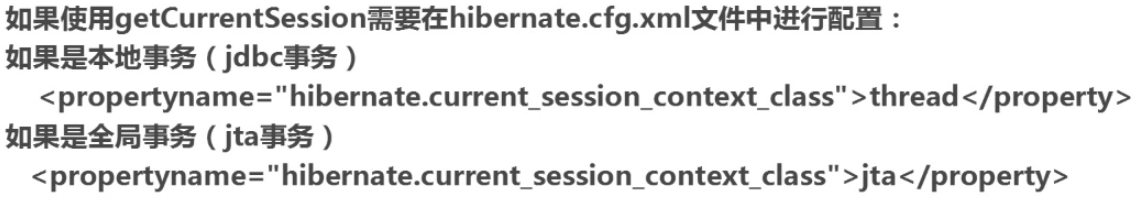


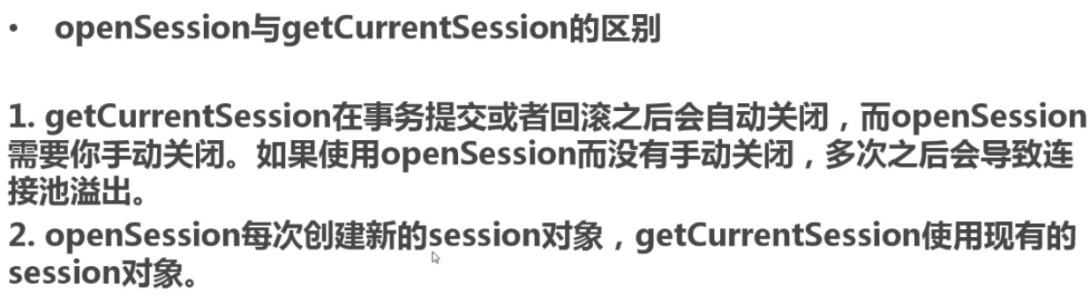


**---------------------------------------------------------------------------------------------**

**4.session详解**

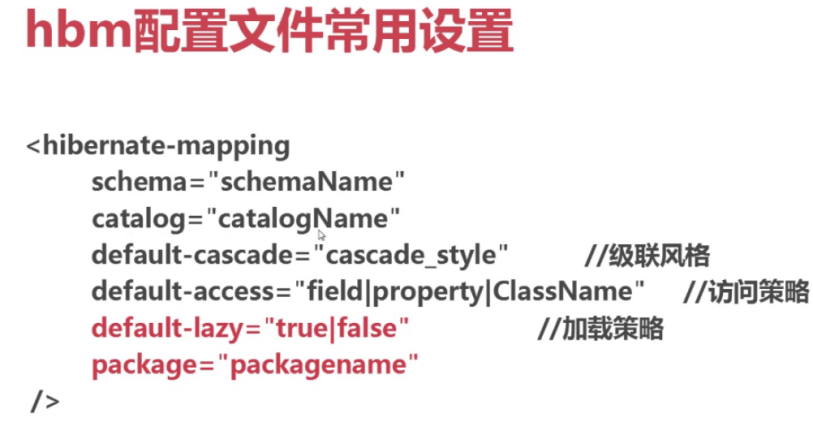
这两种方法都来自sessionFactory



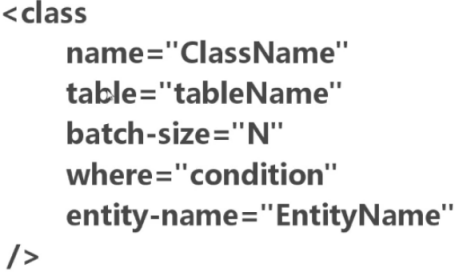


**----------------------------------------------------------------------------------------------**

**5.对象关系映射常用配置**



Schema：模式的名字 catalog：目录名



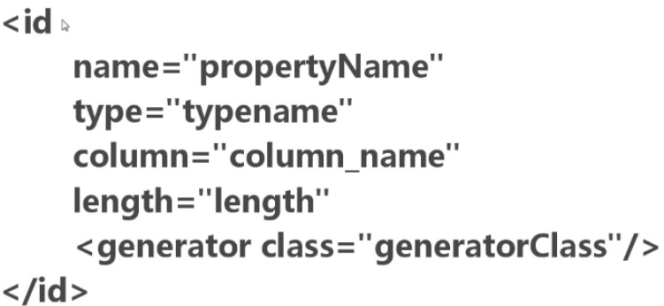
Name：表示对象关系映射，映射的是哪个类

Table：表示把这个类映射成数据库中的哪张表，对应的是他的表名

Batch-size：抓举策略，一次抓举多少条记录

Where：写抓举条件

Entity-name：支持映射多张表，一般一个类对应一张表



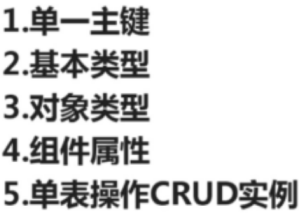
**ID标签：表的一个主键**

Name：要映射的属性 type：数据类型 column：映射成字段的名称 length：指定长度

Generator class：id的子标签，主键生成策略

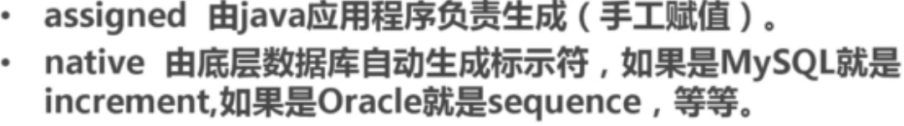


**三．Hibernate单表操作**



**-----------------------------------------------------------------------------------------------**

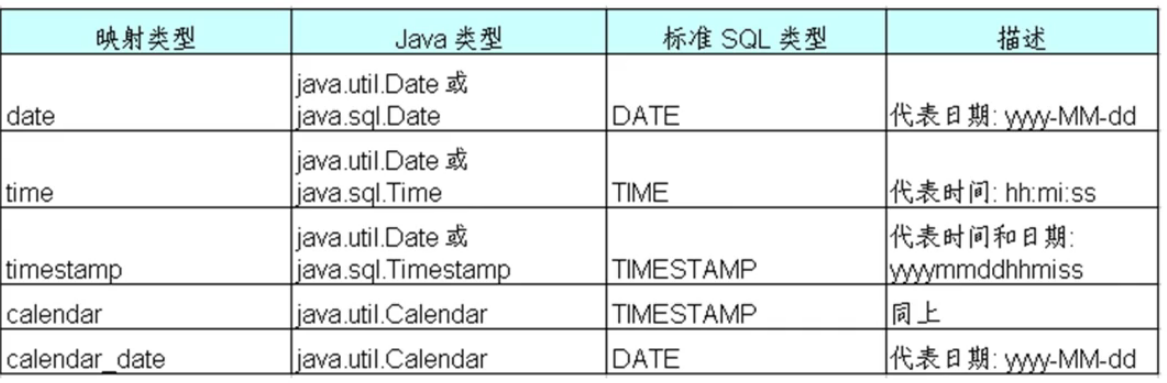
**1.单一主键 生成策略 详见5.1.1\_Hibernate\_Base**



1. **基本类型**

**一般仅适用hibernate映射类型和Java类型**





**3.对象类型**



Binary对应的是字节数组

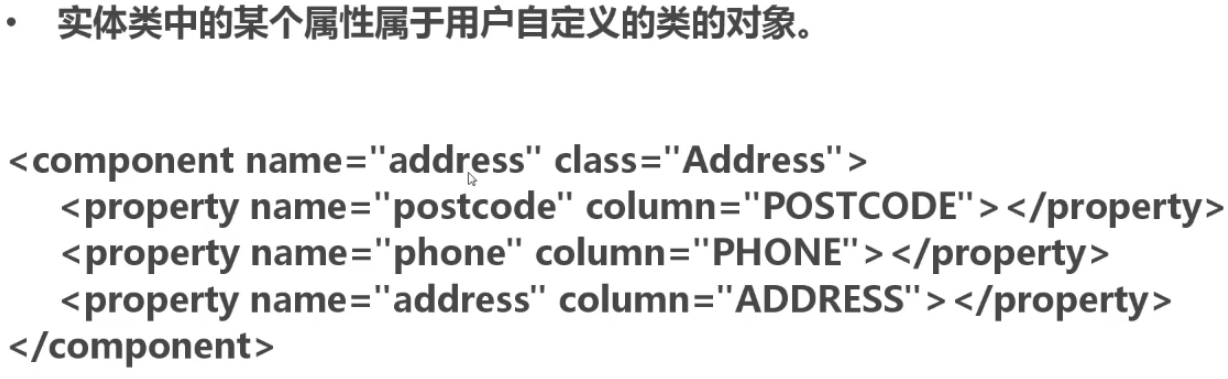
Text和clob对应的都是大文本数据类型

Blob对应的是二进制数据类型，音频视频图片

1. **组件属性**

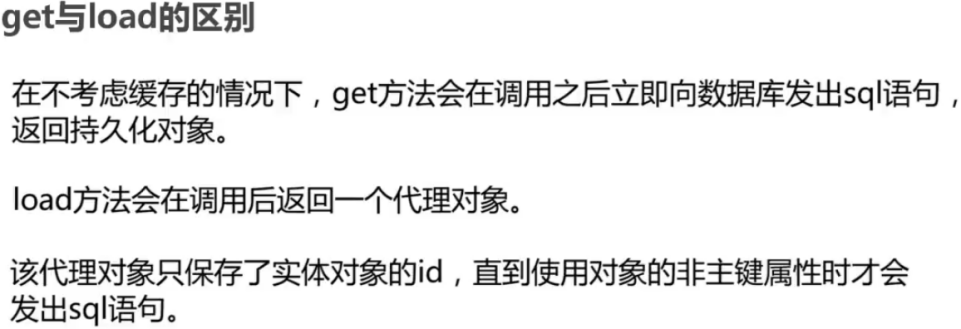
实体类中的某个属性属于用户自定义的类的对象

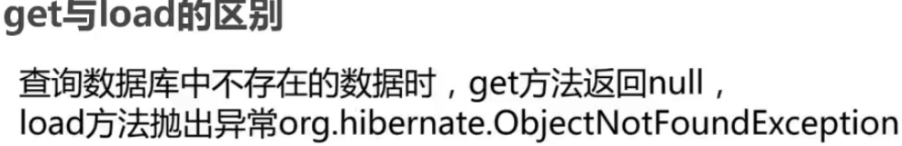
使用组件属性一定要在映射文件中指定所属包名 否则会报错



**5.单表操作CRUD**





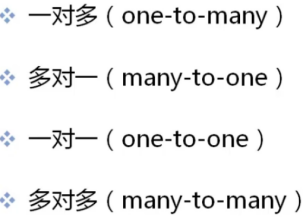


**三．Hibernate一对多映射**

**单值映射：目标角色的基数等于1 一对一，多对一**

**集合映射：目标角色的基础大于1 一对多，多对多**

**关联映射类型：前两种常用重要**



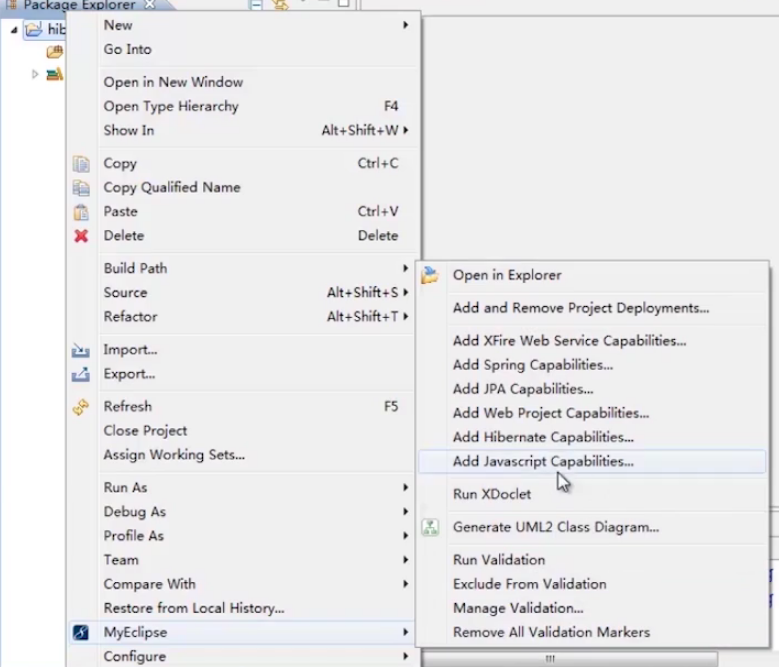
**映射方向分为单项和双向**





**MyEclipse的hibernate配置：**

自动：

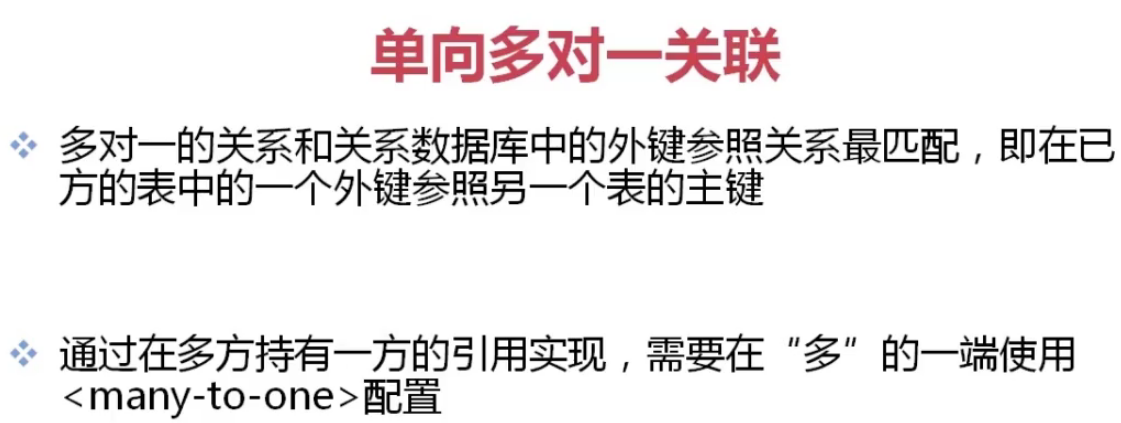


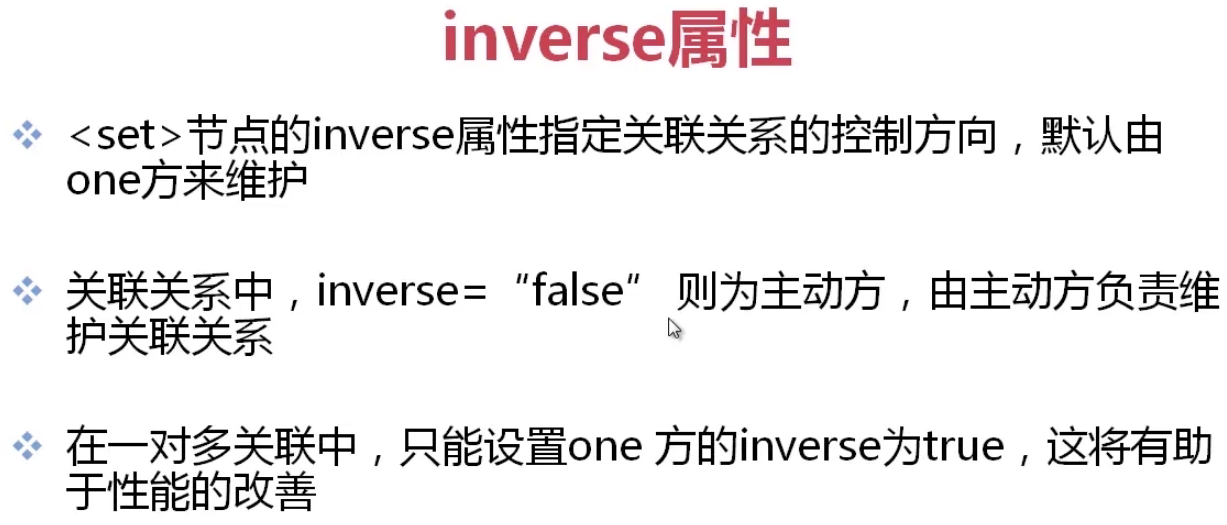
手动：

Hibernate官网下载hibernate-release-4.3.1.final-lib-required中的所有jar包+MySQL驱动包

**案例补充：**

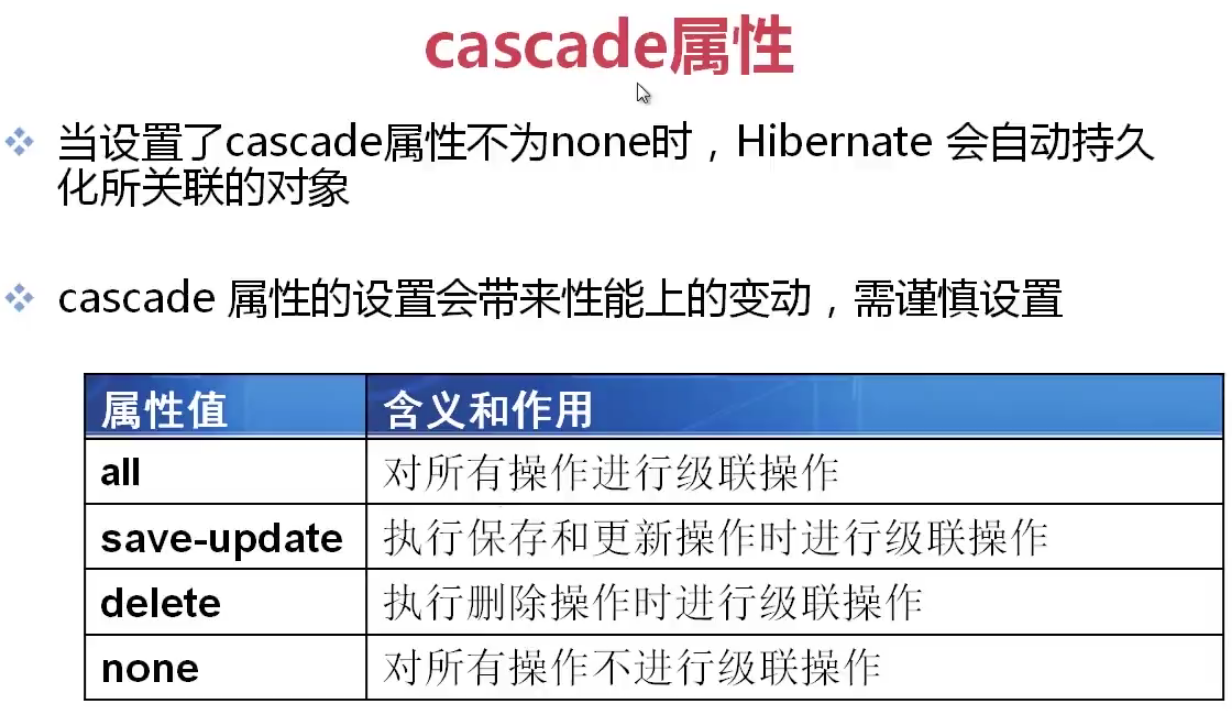






表示反转

**inverse 在 双向 多对一中 写在 单方 的set里 有助于改善性能**



**三．Hibernate-HQL数据查询基础**

HQL：Hibernate Query Language ,Hibernate查询语言

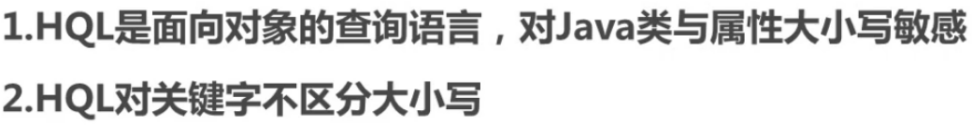
HQL：是面向对象的查询语言，查询主体是映射配置的持久化类及其属性

SQL：查询主体是数据库表

HQL与SQL形式相似，本质不同

HQL语句形式

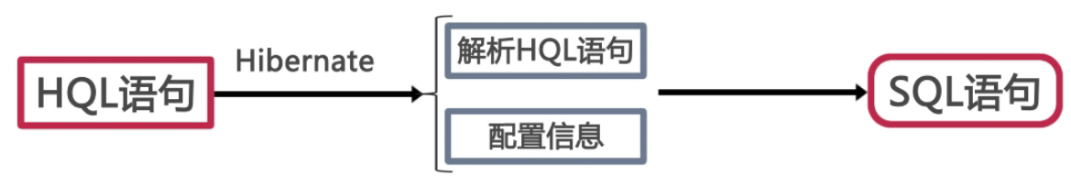




**Query【**org.hibernate.Query**】接口简介：**

1. **Query接口有执行查询的方法**

该方法完成HQL语句的解析和执行过程，并返回查询的结果



1. **Query接口支持方法链编程风格，使得程序代码更为简洁**

方法链编程：调用方法后，返回的结果还是调用方法的这个对象，在调用方法后，直接调用该对象的其他方法，这样可以用一条程序语句完成多个方法的调用和执行；

在Query接口中，使用方法链编程使用最多的场景是查询参数的动态设置，特别是多个参数的设置

1. **Query实例对象的创建**



1. **Query执行查询**



**From子句**

1.HQL语句的最简形式，是唯一不可或缺的字句

原因：from指定了HQL语句查询主体—持久化类及其属性，当编写的HQL语句在执行from子句的情况下，hibernate默认查询该持久化类的所有实例以及该持久化类映射配置的所属信息，当hibernate框架把HQL语句解析成SQL语句之后，就会查询该持久化类映射的数据表中所有的映射字段信息，并将返回的查询结果封装为该持久化类的对象集合

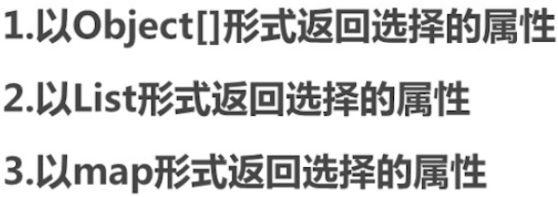
1. 不需要引入持久化类的全限定名，直接引入类名

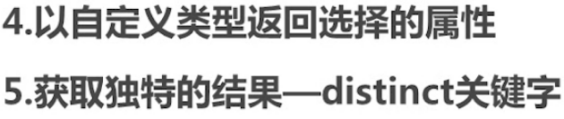
全限定名com.xxx.model.Seller

之所以可以省略包名，是因为auto-import（自动引入）的缺省情况，在hibernate框架中，解析HQL语句时，会根据映射配置信息，自动完成持久化类的导入，

**Select子句**

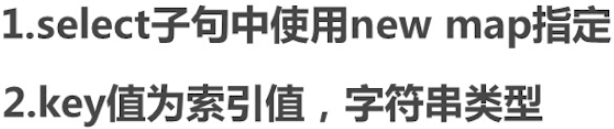
在select子句中，指定选择的对象和属性之后，以哪种数据类型返回执行查询的结果











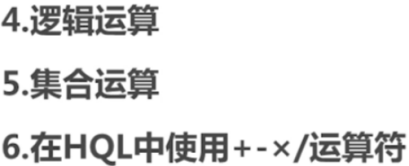
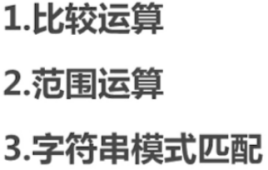
**持久化类中【玛德其实就是实体类】无参构造方法的必要性：**

因为在HQL语句中，当指定了查询目标【即from之后的持久化类】，并且指定相应的构造器【select之后】，hibernate会调用指定的构造器进行对象的创建和数据的封装，如果没有指定相应的构造器，会使用默认的无参构造

String hql = " select new Seller(s.name,s.tel,s.address) from Seller s ";

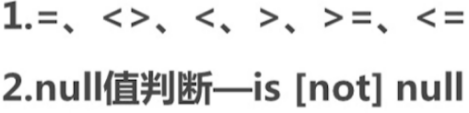
**Where子句**

其实就是一个逻辑表达式，用来设置查询的条件，限制返回的查询结果





**比较运算符，就是将持久化类的属性与给定的条件进行比较**



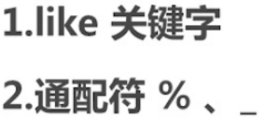


**范围运算**



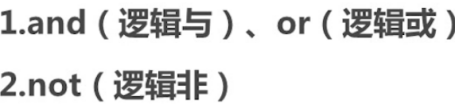
In的候选值列表可以直接列出，也可以是一个子查询

**字符串模式匹配**

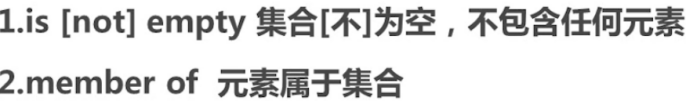


**逻辑运算**



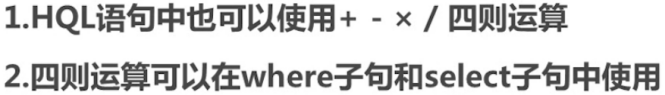


**集合运算**



hibernate转换

**四则元算**



**查询单个对象**



查询结果返回的是一个实例对象

**Order by**

