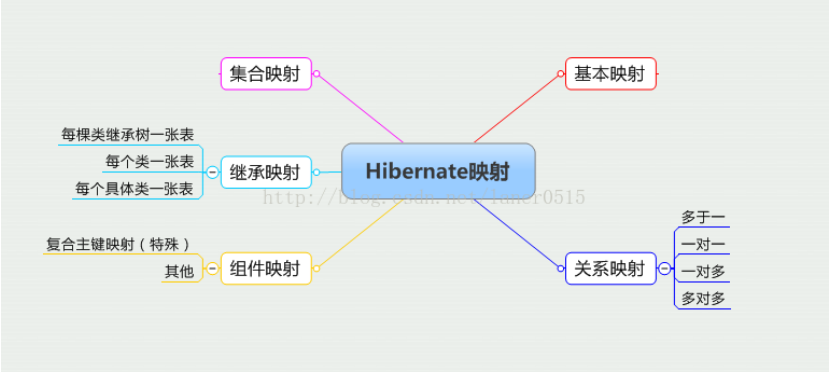
Hibernate主要实现的映射关系：



一对一关联映射

一对一主键单向关联： PERSON——>ADDRESS

```java
public class Person {
    private int personid;
    private String name;
    private int age;
    //在Person对象中有Address对象的关联信息
    private Address address;}
```

```java
public class Address{
    //Address对象中没有Person对象的关联信息
    private int addressid;
    private String addressdetail;}
```

Person.hbm.xml【PK】

```xml
<hibernate-mapping>
    <class name="com.entity.Person" table="PERSON">
        <id name="personid" column="presonid">
            <!--基于主键关联时，主键生成策略是foreign，表明根据关联类生成主键-->
            <generator class="foreign">
                <!--关联持久化类的属性名-->
                <param name="property">address</param>
            </generator>
        </id>
        <property name="name"/>
        <property name="age"/>
        <!--constrained设定为true，表示主键必须与Person中对应的主键相同。-->
        <one-to-one name="address" constrained="true"/>
    </class>
</hibernate-mapping>
```

Person.hbm.xml【FK】

```xml
<hibernate-mapping>
        <class name="com.entity.Person" table="PERSON">
                <id name="personid"><generator class="identity"/> </id>
                <property name="name"/>
                <property name="age"/>
        <!--用来映射关联PO column是Address在该表中的外键列名,增加unique变成唯一的-->
                <many-to-one name="address" unique="true"/>
        </class>
</hibernate-mapping>
```

外键关联和主键关联不同的地方是采用<many-to-one>标签来映射，一对一唯一外键关联映射其实是多对一的特例。<many-to-one>指定多的一端unique为true，这样就限制了多的一端的多重性为一，就是这样来映射的。

一对一主键双向关联：PERSON<——>ADDRESS

```java
public class Person implements java.io.Serializable {

  private Long id;
  private String name;
  //双向关联中Person对象中有Adderss对象的关联信息
  private Address address;
```

```java
public class Address implements java.io.Serializable {
  private Long id;
  //Adderss对象中也有Person对象的关联信息
  private Person person;
  private String detail;
```

Person.hbm.xml【PK】

```xml
<hibernate-mapping>
  <class name="entity.Person" table="person">
    <id name="id" type="java.lang.Long">
      <column name="id" />
      <generator class="identity" />
    </id>
    <property name="name" type="java.lang.String">
      <column name="name" length="24" not-null="true">
        <comment>姓名</comment>
      </column>
    </property>
    <one-to-one name="address"/>
  </class>
</hibernate-mapping>
```

Address.hbm.xml【PK】

```xml
<hibernate-mapping>
  <class name="entity.Address" table="address" catalog="mydb">
    <id name="id" type="java.lang.Long">
      <column name="id" />
      <!-- class="foreign": 一对一主键映射中，使用另外一个相关联的对象的标识符 -->
      <generator class="foreign">
        <param name="property">person</param>
      </generator>
    </id>
   <property name="detail" type="java.lang.String">
      <column name="detail" length="120" not-null="true">
        <comment>详细地址</comment>
      </column>
    </property>
    <!-- 表示在address表存在一个外键约束，外键参考相关联的表person -->
    <one-to-one name="person" constrained="true" />
  </class>
</hibernate-mapping>
```

Person.hbm.xml【FK】

```
<hibernate-mapping>
  <class name="com.entity.Person" table="person">
    <id name="personid" type="java.lang.Long">
      <column name="personid" />
      <generator class="identity" />
    </id>
    <property name="name" type="java.lang.String">
      <column name="name" length="24" not-null="true">
        <comment>姓名</comment>
      </column>
    </property>
   <!--双向关联配置-->
    <one-to-one name="address" />
  </class>
</hibernate-mapping>
```

Address.hbm.xml【FK】

```
<hibernate-mapping>
  <class name="com.entity.Address" table="address" catalog="testdb">
    <id name="addressid" type="java.lang.Long">
      <column name="addressid" />
      <generator class="identity" />
    </id>
    <property name="detail" type="java.lang.String">
      <column name="detail" length="120" not-null="true">
        <comment>详细地址</comment>
      </column>
    </property>
    <many-to-one name="person" class="entity.Person" unique="true">
      <column name="personid">
        <comment>人的ID</comment>
      </column>
    </many-to-one>
  </class>
</hibernate-mapping>
```

注意：因为一对一的主键关联映射扩展性不好，当我们的需要发生改变想要将其变为一对多的时候变无法操作了，所以我们遇到一对一关联的时候经常会采用唯一外键关联来解决问题，而很少使用一对一主键关联。

一对多关联映射

一对多单向关联：CLASSES——>STUDENT

```
public class Classes {
    private int id;
    private String name;
//Set支持延迟加载因为多个学生所以我们用Set集合关联
    private Set students;   }
```

```
public class Student {
    private int id;
    private String name;
}
```

Classes中的set属性只是说明了lazy load，并没有为属性配置对应的对象，要在映射文件中配置，在set标签中添加<one-to-many>标签

Classes.hbm.xml

```
<hibernate-mapping>
    <class name="com.hibernate.Classes" table="t_classes">
        <id name="id">
            <generator class="native"/>
        </id>
        <property name="name"/>
        <set name="students">
            <key column="classesid"></key>
            <one-to-many class="com.hibernate.Student"></one-to-many>
        </set>
    </class>
</hibernate-mapping>
```

一对多双向关联：CLASSES<——>STUDENT

```
public class Classes {
    private int id;
    private String name;
    //Set支持延迟加载
    private Set<Student> students;
}
```

```
public class Student {
    private int id;
    private String name;
    //添加class对象关联信息因为是一方所以我们用一个对象关联
    private Classes classes;     }
```

Classes.hbm.xml

同上

Student.hbm.xml

```
<hibernate-mapping>
    <class name="com.hibernate.Student" table="t_student">
        <id name="id">
            <generator class="native"/>
        </id>
        <property name="name"/>
        <!-- 在many的一端中添加Classes列,并且name要和Classes.hbm.xml的列名相同-->
        <many-to-one name="classes" column="classesid"></many-to-one>
    </class>
</hibernate-mapping>
```

因为Student一方是多方对应的Classes是一方 所以我们要用<many-to-one>来关联一方

多对多关联映射

多对多单向关联：TEACHER——>STUDENT

```
public class Teacher {
    private int id;
    private String name;
    private Set<Student> students = new HashSet<Student>();
}
```

```
public class Student {
    private int id;
    private String name;
    private String title;
}
```

Teacher.hbm.xml

```
<hibernate-mapping>
    <class name="com.hibernate.Teacher" table="t_teacher">
        <id name="id">
            <generator class="native"/>
        </id>
        <property name="name"/>
        <!--生成一张新表存放两个关联对象的ID-->
        <set name="students" table="Teacher_Sutdent">
            <!--将Teacher表的外键关联 注意不是对象的属性是表中的字段-->
            <key column="teacher_id"></key>
            <!--将Student表的外键关联 注意不是对象的属性是表中的字段-->
            <many-to-many class="com.hibernate.Student"
column="student_id"></many-to-many>
        </set>
    </class>
</hibernate-mapping>
```

使用<many-to-many>标签，并且在标签中添加上对应的列关系，因为你要让两个对象中都要清楚它们之间的映射是如何使用的，并且在生成的关系表中哪一列是对应的自己的外键，所以要在该标签中指明，另外在<set>标签中添加table属性会指明要生成新表

多对多双向关联：TEACHER<——>STUDENT

```
public class Teacher {
    private int id;
    private String name;
    private Set<Student> students =
new HashSet<Student>();
}
```

```
public class Student {
    private int id;
    private String name;
    private String title;
    private Set<Teacher> teachers =
new HashSet<Teacher>();
}
```

Teacher.hbm.xml同单向多对多一样故省略

同上

Student.hbm.xml

```
<hibernate-mapping>
    <class name="com.hibernate.Student" table="t_student">
        <id name="id">
            <generator class="native"/>
        </id>
        <property name="name"/>
        <!--生成一张新表存放两个表的Id-->
        <set name="teachers" table="Teacher_Student">
            <!--将Teacher表的外键关联 注意不是对象的属性是表中的字段-->
            <key column="student_id"></key>
            <!--将Student表的外键关联 注意不是对象的属性是表中的字段-->
            <many-to-many class="com.hibernate.Teacher"
column="teacher_id"></many-to-many>
        </set>
    </class>
</hibernate-mapping>
```

多对一关联映射

对比一对一关联映射和多对一唯一外键关联映射，其实它们两个都是使用了<many-to-one>本质上都是外键约束，只不过一对一的是唯一映射，需要添加unique="true"的属性，其它的它们两个是相同的。

多对一单向关联：

```
public class Department {
    private int id;
    private String name;
}
```

```
public class Employee {
    private int id;
    private String name;
    private Department depart;
}
```

一个部门对应多个员工，这里是以部门的对象来作为员工的属性的

Employee.hbm.xml

```
<hibernate-mapping package="com.suo.domain">
    <class name="Employee">
        <id name="id">
            <generator class="native"/>
        </id>
        <property name="name"/>
        <many-to-one name="depart"></many-to-one>
        <!-- many-to-one指明了外键 ，会根据反射机制，找到要和Employee建立多对一关系的类，该列默认的是可以为空的-->
    </class>
</hibernate-mapping>
```

多对一双向关联：

```
public class Department {
    private int id;
    private String name;
    private Set<Employee> emps;}
```

```
public class Employee {
    private int id;
    private String name;
    private Department depart;    }
```

Departement .hbm.xml

```
<hibernate-mapping package="com.suo.domain">
    <class name="Department">
        <id name="id">
            <generator class="native"/>
        </id>
        <property name="name"/>
        <set name="emps">
            <key column="depart_id"/><!-- key指明了员工表中的外键-->
            <one-to-many class="Employee"/><!-- one-to-many指明了和哪个类进行一对多的映射 -->
        </set>
        <!--用set标签表示Department中的员工集合的属性，这个属性并没有映射到数据库中的部门表中，即部门表中，并没有emps这样的一个列。  -->
    </class>
</hibernate-mapping>
```

Employee.hbm.xml同单向关联配置相同故省略

级联操作 Cascade：

一.简单的介绍

CASCADE用来说明当对主对象进行某种操作时是否对其关联的从对象也作类似的操作

常用的cascade: none, all, save-update, delete, lock, refresh, evict, replicate, persist, merge, delete-orphan(one-to-many)。

一般对many-to-one, many-to-many不设置级联, 在<one-to-one>和<one-to-many>中设置级联。

INVERSE表示“是否放弃维护关联关系”(在JAVA里两个对象产生关联时，对数据库表的影响)

在one-to-many和many-to-many的集合定义中使用，inverse=”true”表示该对象不维护关联关系；该属性的值一般在使用有序集合时设置成false（注意hibernate的缺省值是false）。one-to-many维护关联关系就是更新外键。many-to-many维护关联关系就是在中间表增减记录。

注: 配置成one-to-one的对象不维护关联关系

三。代码练习

```
<set name="emps" cascade="save-update">
 <key column="depart_id"/>
  <one-to-many class="Employee"/>
</set>
<set name="students" table="taacher_student" inverse="true"><!-- table是用来指定中间表的属性 -->
<key column="teacher_id"></key><!-- 查找教师id时，链接中间表表的teacher_id -->
<many-to-many class="Student" column="student_id"></many-to-many>
</set>
```

关系映射总结：

单向关联	
一对一主键关联	在有关联信息配置文件里加入<one-to-one constrained=“true”>，将constrained设为true。表示的主键必须与这个对象中对应resource的主键相同
一对一外键关联	在有关联信息一方的配置文件里加入<many-to-one unique="true">并且将unique属性设置为true 表示这个主键是唯一的
一对多单向关联	在有关联信息一方的配置文件里加入<set>在<set>中加入<one-to-many/> <set> <key column="关联的外键"> <one-to-many/> </set>
多对多单向关联	在有关联信息一方的配置文件里加入<set>生成一张新表用来存放两个表的外键> <set table=""> <key column="当前表的外键ID"/> <many-to-many class="关联对象路径" column="关联对象表的Id"> </set>
多对一单向关联	单向关联时我们需要在有关联信息一方的配置文件里加入<many-to-one>

双向关联	
一对一主键关联	在从表的一方加入<one-to-one constrained="true"> 还需要在主表加入<one-to-one>
一对一外键关联	除了在从表中加入<many-to-one unique="true"> 也需要在主表加入<one-to-one>
一对多双向关联	除了在一方中加入<set><one-to-many></set> 还需要在多放加入<many-to-one>
多对多双向关联	需要在两方都加入<set><many-to-many></set> 注：<set>中的table="表名"　表明两方的配置要一样 **<set name="关联对象的属性名" table="生成一张新表">** **<key column="当前对象数据库表的外键"/>** **<many-to-many class="关联对象的类路径" column="关联对象数据库表的外键">** **</set>**
多对一双向关联	要在多方中加入**<many-to-one>**还要在一方中加入**<set>** **<set>** **<key column="关联外键"/>** **<one-to-many>** **</set>**