1. 根据id的查询（get或load方法）

2、Hql(Hibernate Query Language)：

关系型数据库的表名换成类名-----列名换成成员变量的名字

如：

Query q=session.createQuery(“from Customer c where c.name=:name and c.age=:age”);

q.setString(“name”,”aaa”);---------------------q.setParameter("name","aaa");

q.setInteger(“age”,23);

List result = q.list();

(参数绑定的两种方式，也可以使用问号) 索引从零开始

可以写成一条语句：

List result = session.createQuery(“from Customer c where c.name=:name”+”and c.age=:age”).setString(“name”,”aaa”).setInteger(“age”,23).list();

当能确定是只返回一条时：

User user = session.createQuery(hql).setParameter("userName","zhangsan").uniqueResult();

当使用hql语句查询数据 做了投影时，将不会返回对象，将返回Object数组

可以这样来解决这个问题：

String hql = "select new domain.User(u.name,u.age) from User u where u.name like :uName";

查询结果排序：

Query q=session.createQuery(“from User u order by u.name desc”);

关于分页查询：

q.setFirstResult(0);

q.setMaxResults(10);

查询单个对象：

Customer c=(Customer)session.createQuery("from User u order by u.name asc”).setMaxResults(1).uniqueResult();

使用setParameter方法绑定任意类型的参数：

Query q=session.createQuery("from Order o where o.customer=:customer"+"and o.orderNumber like :orderNumber");

q.setParameter("customer",customer,Hibernate.entity(Customer.class));

q.setParameter("orderNumber",orderNumber,Hibernate.STRING);

使用setProperties方法绑定参数：

Customer c=new Customer();

c.setName(“zhangsan”);

c.setAge(33);

Query q=session.createQuery(“from Customer c where c.name=:name and c.age=:age”);

q.setProperties(c);

在配置文件中编写hql语句

<hibernate-mapping>

<class name=“mypack.Customer” table=“CUSTOMERS”>

</class>

<query name=“aFind”><![CDATA[

from Customer c where c.name like :name]]></query>

</hibernate-mapping>

Query q=session.getNamedQuery(“aFind”);

q.setString(“name”,name);

List result=q.list();

String hql = "select distinct u.id from User u join u.orders o group by u.id having sum(o.price)>:price";

Query q = s.createQuery(hql);

q.setParameter("price", price);

List list = q.list();

String hql2 = "select \* from User u where u.id in (:userIds)";

Query q2 = s.createQuery(hql2);

q2.setParameter("userIds", list);

String hql = "from User u where u.orders is not empty";

1. QBC（Query By Criteria）

Criteria c=session.createCriteria(Customer.class);

SimpleExpression c1=Restrictions.like(“name”,”zhang%”);

SimpleExpression c2=Restrictions.eq(“age”,new Integer(99));

c=c.add(c1).add(c2);

List result=c.list();

查询结果排序

Criteria c=session.createCriteria(Customer.class);

c.addOrder(Order.asc(“name”));

c.addOrder(Order.desc(“age”));

关于分页查询：

c.setFirstResult(0);

c.setMaxResults(10);

Criteria中的查询结果比较

检索年龄大于18的Customer：

Criteria c=session.createCriteria(Customer.class);

c.add(Restrictions.gt(“age”,18));

检索年龄不等于18的Customer：

c.add(Restrictions.not(Restrictions.eq(“age”,new Integer(18))));

检索姓名为空的Customer：

c.add(Restrictions.isNull(“name”));

检索不属于任何客户的订单：

c.add(Restrictions.isNull(“customer”));

检索名字为zhangsan的客户：

c.add(Restrictions.eq(“name”,”zhangsan”).ignoreCase());

ge---------------------->=

le----------------------<=

Criteria中的OR

Restrictions.or(Restrictions.eq(),Restrictions.or(Restrictions.between(),Resctions.like()))

String[] names={“Tom”,”Mike”,”Jack”};

c.add(Restrictions.in(“name”,names));

检索年龄在18到30之间的客户：

c.add(Restrictions.between(“age”,new Integer(18),new Integer(30));

检索年龄不在18到30的客户：

c.add(Restrictions.not(Restrictions.between(“age”,new Integer(18),new Integer(30)));

Critieria c = s.createCriteria(Order.class);

c.add(Restrictions.eq("customer",uu));

return c.list();

Count和分组

Critieria c = s.createCriteria(Customer.class)

c.setProjection(Projections.rowCount());

c.setProjection(Projections.groupProperty("sex"));

//Select count(\*) from customer group by sex

Critieria c = s.createCriteria(Customer.class).setProjection(Projections.ProjectionList().add(Projections.rowCount()).add(Projections.groupProperty("sex")))

c.setProjection(Projections.rowCount());

c.setProjection(Projections.groupProperty("sex"));

Select count(\*) from customer group by sex;

1. QBE（Query By Example）

Customer c=new Customer();

c.setAge(99);

c.setName("aaa");

List result=session.createCriteria(Customer.class).add(Example.create(c)).list();

QBE只支持＝和like，不支持or > <之类的查询

1. SQL

String sql = "select \* from users u where u.name like :name";

Query q = s.createSQLQuery(sql).addEntity(User.class);

q.setParameter("name", name);