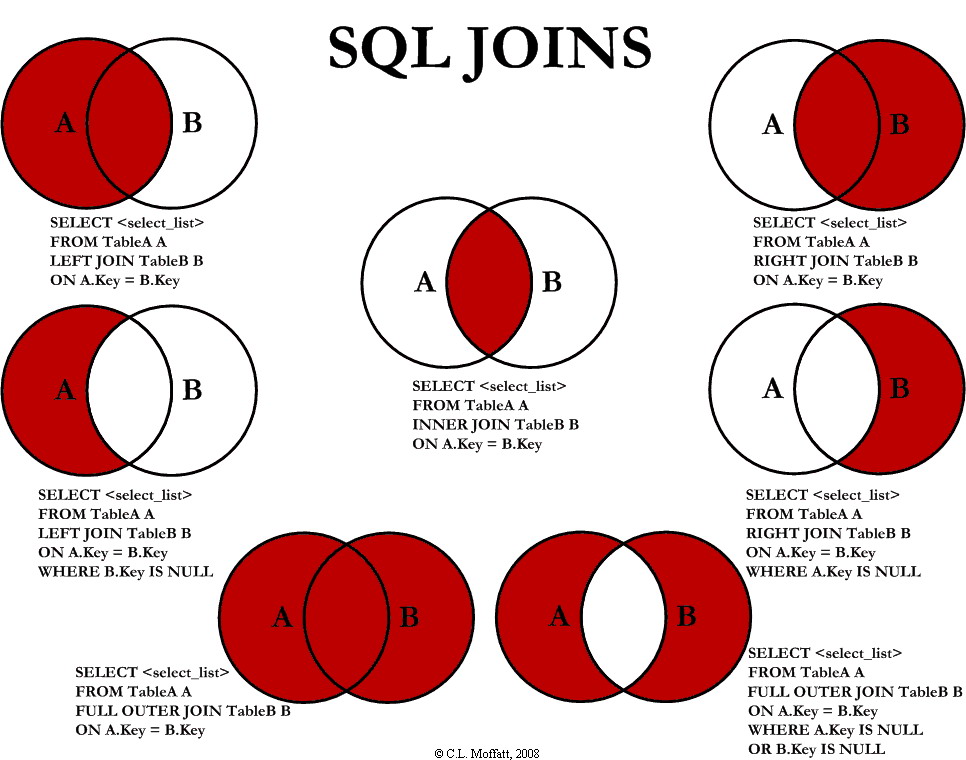
Hibernate supports 4 types of joins..

1. Left Join or left outer join
2. Right  Join or right outer join
3. Full Join or full outer join
4. Inner Join or join

the DEFAULT join in hibernate is Inner join

* Left join means, the objects from both sides of the join are selected and more objects  from left side are selected, even though no equal objects are there at right side
* Right join means equal objects are selected from database and more objects are from right side of join are selected even though there is no equal objects are exist left side
* Full join means, both equal and un-equal objects from both sides of join are selected
* Inner join means only equal objects are selected  and the remaining are discarded
* At the time of construction the join statements, we need to use the properties created in pojo class to apply relationship between the objects

Example Joins using SQL:



HQL Joins Syntax:

select a1.<VN>,a2.<VN>

from <Classname> <aliasesName1>

[join type] <aliasesName1>.<DVN> as <aliasesName2>

[where...]

VN: Variable Name

DVN: dependency variable Name

Example:

select e.empName,addr.loc

from com.app.Employee e left join e.addr as addr;

consider below Example:

public class Customer {

    private String name;

    private long custmerId;

    private Set<Policy> policyies = new HashSet<Policy>(0);

...

}

public class Policy {

    private long policyId;

    private String policyName;

...

}

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-mapping PUBLIC

    "-//Hibernate/Hibernate Mapping DTD 3.0//EN"

    "<http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd>">

<hibernate-mapping>

    <class name="com.app.Customer" table="customer">

                <id name="custmerId" type="long" column="custmerId">

                </id>

        <property name="name" />

        <set name="policyies" table="customer\_policy" cascade="all" >

            <key column="custmer\_Id" />

            <many-to-many column="policy\_Id" class="com.app.Policy" />

        </set>

    </class>

<class name="com.app.Policy" table="policy">

        <id name="policyId" type="long" column="policyId">

        </id>

        <property name="policyName" />

    </class>

</hibernate-mapping>

Hibernate Util :

public class HibernateUtil {

    private static final SessionFactory sessionFactory;

    static {

        try {

            sessionFactory = new Configuration().configure()

                    .buildSessionFactory();

        } catch (Throwable ex) {

            System.err.println("SessionFactory creation failed" + ex);

            throw new ExceptionInInitializerError(ex);

        }

    }

    public static SessionFactory getSessionFactory() {

        return sessionFactory;

    }

}

Client Prgm:

public class Main {

    public static void main(String[] args) {

        Session session = HibernateUtil.getSessionFactory().openSession();

        Transaction transaction = null;

        try {

            transaction = session.beginTransaction();

            Set<Policy> policySet = new HashSet<Policy>();

            policySet.add(new Policy("LIC"));

            policySet.add(new Policy("HDFC PLAN"));

            Customer c1=new Customer("AAA");

            c1.setPolicyies(policySet);

            Customer c2=new Customer("SAM");

            c2.setPolicyies(policySet);

            session.save(c1);

            session.save(c2);

            transaction.commit();

            System.out.println("Records inserted sucessessfully");

getPolocy();

        } catch (HibernateException e) {

            transaction.rollback();

            e.printStackTrace();

        } finally {

            session.close();

        }

    }

    public static void getPolocy()

    {

        Session session = HibernateUtil.getSessionFactory().openSession();

        Transaction       transaction = session.beginTransaction();

                        List policy = session.createQuery("from com.app.Customer c left join c. policy as p").list();

                        sysout(policy);

}