一，一对一的关联关系（***）

注意事项：

①在存储数据到数据库中的时候，尽量避免使用中文，因为中文可能会存在乱码。

②在mybatis中，任何数据都不要使用基本数据类型，都使用引用数据类型。有时候会报错。

③千万记住实体类需要为其设置一个set和get方法，否则mybatis可能不能将其映射到数据库中。

④一对一关联关系中，需要在实体类中将对方的一个对象作为该类的一个属性。

⑤如果在mybatis中使用实体类对象接收查询结果的返回值，并且该对象中的一个属性刚好是另一个实体类对象的一个实例，需要对其进行重新配置。

1，局部配置文件中的详细配置信息：

```
<!-- resultMap表示结果映射：type表示数据返回的一个数据类型，id表示一个别名，里面的一个result表示实体类中的一个属性
     -->
    <resultMap type="cn.dao.userEntity.Husband" id="hus">
    <!--
    property:代表的是实体类中的属性；
    javaType：代表的是实体类中的属性名的类型；
    column：数据中查询语句的查询字段名，有别名的表示别名；
    jdbcType：数据库中查询语句字段名的数据类型
     -->
        <result property="id" javaType="Long" column="id" jdbcType="NUMERIC" />
        <result property="husbandName" javaType="String" column="husbandName"
            jdbcType="VARCHAR" />
        <result property="sex" javaType="String" column="sex" jdbcType="VARCHAR" />
        <result property="wifeId" javaType="Long" column="wifeId"
            jdbcType="NUMERIC" />
        <!-- 程序员自己定义的类需要使用另一个标签 -->
        <!-- jdbcType表示mysql数据库中对应字段的数据类型，property后面 值表示实体类中对应的
        属性名，JavaType表示该属性在实体类中的数据类型是什么样的。column表示对应的实体在表中的列名
        。其中jdbcType可以省略不写。
        -->
        <association property="wife" javaType="cn.dao.userEntity.Wife">
            <result property="id" javaType="Long" column="id" jdbcType="NUMERIC" />
            <result property="wifeName" javaType="String" column="wifeName"
                jdbcType="VARCHAR" />
            <result property="sex" javaType="String" column="sex"
                jdbcType="VARCHAR" />
        </association>
    </resultMap>
    <!-- resultMap:表示上面resultMap中id的值 -->
    <select id="One2OneTest" resultMap="hus">
        select h.*,w.wifeName,w.sex
        from husband h inner join wife w on
        h.wifeId=w.id;
    </select>
```

java中Junit代码：

```
@Test
public void One2OneTest(){
    // select h.*,w.wifeName,w.sex from husband h inner join wife w on h.wifeId=w.id;
    List<Husband> maplist = session.selectList("cn.dao.userDaoImpl.userDao.One2OneTest");
    for (Husband map : maplist) {
        System.out.println(map);
    }
}
```

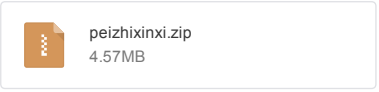
二，逆向工程

（https://www.jianshu.com/p/39f90e81ee35）

1，使用逆向工程，实体类什么的就不需要了，也就是再无dao层。

2，逆向工程的解决办法

3，具体的写法如下：



4，generatorConfig.xml配置文件的配置信息：

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE generatorConfiguration
 PUBLIC "-//mybatis.org//DTD MyBatis Generator Configuration 1.0//EN"
 "http://mybatis.org/dtd/mybatis-generator-config_1_0.dtd">

<generatorConfiguration>
    <context id="testTables" targetRuntime="MyBatis3">

        <commentGenerator>
            <property name="suppressAllComments" value="true" />
        </commentGenerator>
        <!-- 连接数据库 -->
        <jdbcConnection driverClass="com.mysql.cj.jdbc.Driver"
            connectionURL="jdbc:mysql://localhost:3306/db_mybatis?
useSSL=false&serverTimezone=UTC&characterEncoding=utf8"
            userId="root" password="AQL271422">
        </jdbcConnection>

        <javaTypeResolver>
            <property name="forceBigDecimals" value="false" />
        </javaTypeResolver>
        <!-- targetPackage：生成po类的包名 targetProject：生成po类的位置 -->
        <javaModelGenerator targetPackage="com.mybatis.po"
            targetProject="H:\mybatis_study\mybatis_learning\niXiangGongCheng\code">
            <property name="enableSubPackages" value="true" />
            <property name="trimStrings" value="true" />
        </javaModelGenerator>

        <!-- targetPackage：mapper接口文件生成的包名 targetProject：mapper接口文件生成的项目位置 -->
        <sqlMapGenerator targetPackage="com.mybatis.dao"
            targetProject="H:\mybatis_study\mybatis_learning\niXiangGongCheng\code">
            <property name="enableSubPackages" value="true" />
        </sqlMapGenerator>

        <!-- targetPackage：mapper映射文件生成的包名 targetProject：mapper映射文件生成的项目位置 -->
        <javaClientGenerator type="XMLMAPPER"
            targetPackage="com.mybatis.dao"
targetProject="H:\mybatis_study\mybatis_learning\niXiangGongCheng\code">
            <property name="enableSubPackages" value="true" />
        </javaClientGenerator>
        <!-- 配置需要生成的表 ，表中的一条记录映射着实体类中的一个实体。一个table表示数据库中的一张实际存在的表，也就映射
在java中的一个实体中-->
        <table tableName="users" domainObjectName="user"></table>
    </context>
</generatorConfiguration>
```

5，log4j.properties配置文件

```
# Global logging configuration
log4j.rootLogger=DEBUG, stdout
# Console output...
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%5p [%t] - %m%n
```

\u4F5C\u8005\uFF1A\u75BE\u884C\u7684\u8717\u725B
\u6765\u6E90\uFF1ACSDN
\u539F\u6587\uFF1Ahttps://blog.csdn.net/qq_42262803/article/details/86485140
\u7248\u6743\u58F0\u660E\uFF1A\u672C\u6587\u4E3A\u535A\u4E3B\u539F\u521B\u6587\u7AE0\uFF0C\u8F6C\u8F7D\u8BF7\

6，java的逆向工程生成代码

```
package nixianggongcheng;
import java.io.File;
import java.util.ArrayList;
```

```
import java.util.List;
import org.mybatis.generator.api.MyBatisGenerator;
import org.mybatis.generator.config.Configuration;
import org.mybatis.generator.config.xml.ConfigurationParser;
import org.mybatis.generator.internal.DefaultShellCallback;
public class GeneratorSqlMap {
    public void generator() throws Exception{
        List<String> warnings = new ArrayList<String>();
        boolean overwrite = true;
        //指定逆向工程配置文件
        File configFile = new File("generatorConfig.xml");
        ConfigurationParser cp = new ConfigurationParser(warnings);
        Configuration config = cp.parseConfiguration(configFile);
        DefaultShellCallback callback = new DefaultShellCallback(overwrite);
        MyBatisGenerator myBatisGenerator = new MyBatisGenerator(config, callback, warnings);
        myBatisGenerator.generate(null);
    }
    public static void main(String[] args) {
        GeneratorSqlMap generatorSqlMap = new GeneratorSqlMap();
        try {
            generatorSqlMap.generator();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```