代码：

mybatis-config.xml

1. <?xml version="1.0" encoding="UTF-8" ?>
2. <!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN" "http://mybatis.org/dtd/mybatis-3-config.dtd">
3. *<!-- XML 配置文件包含对 MyBatis 系统的核心设置 -->*
4. <configuration>
5. <properties resource="db.properties"/>
6. *<!-- 指定 MyBatis 所用日志的具体实现 -->*
7. <settings>
8. <setting name="logImpl" value="LOG4J"/>
9. </settings>
11. <environments default="mysql">
12. *<!-- 环境配置，即连接的数据库。 -->*
13. <environment id="mysql">
14. *<!-- 指定事务管理类型，type="JDBC"指直接简单使用了JDBC的提交和回滚设置 -->*
15. <transactionManager type="JDBC"/>
16. *<!-- dataSource指数据源配置，POOLED是JDBC连接对象的数据源连接池的实现。 -->*
17. <dataSource type="POOLED">
18. <property name="driver" value="${driver}"/>
19. <property name="url" value="${url}"/>
20. <property name="username" value="${username}"/>
21. <property name="password" value="${password}"/>
22. </dataSource>
23. </environment>
24. </environments>
26. *<!-- mappers告诉了MyBatis去哪里找持久化类的映射文件 -->*
27. <mappers>
28. <mapper class="com.mapper.PersonMapper"/>
29. <mapper class="com.mapper.CardMapper"/>
30. </mappers>
31. </configuration>

Card.java

1. package com.bean;
3. import java.io.Serializable;
5. public class Card implements Serializable {
6. private Integer id; *// 主键id*
7. private String code; *// 身份证编号*
8. public Card() {
9. super();
10. *// TODO Auto-generated constructor stub*
11. }
12. public Integer getId() {
13. return id;
14. }
15. public void setId(Integer id) {
16. this.id = id;
17. }
18. public String getCode() {
19. return code;
20. }
21. public void setCode(String code) {
22. this.code = code;
23. }
24. @Override
25. public String toString() {
26. *// TODO Auto-generated method stub*
27. return "Card [id=" + id + ", code=" + code + "]";
28. }
30. }

Person.java

1. package com.bean;
3. import java.io.Serializable;
5. public class Person implements Serializable {
6. private Integer id; *// 主键id*
7. private String name; *// 姓名*
8. private String sex; *// 性别*
9. private Integer age; *// 年龄*
11. *// 人和身份证是一对一的关系，即一个人只有一个身份证*
12. private Card card;
14. public Person() {
15. super();
16. *// TODO Auto-generated constructor stub*
17. }
19. public Integer getId() {
20. return id;
21. }
23. public void setId(Integer id) {
24. this.id = id;
25. }
27. public String getName() {
28. return name;
29. }
31. public void setName(String name) {
32. this.name = name;
33. }
35. public String getSex() {
36. return sex;
37. }
39. public void setSex(String sex) {
40. this.sex = sex;
41. }
43. public Integer getAge() {
44. return age;
45. }
47. public void setAge(Integer age) {
48. this.age = age;
49. }
51. public Card getCard() {
52. return card;
53. }
55. public void setCard(Card card) {
56. this.card = card;
57. }
59. @Override
60. public String toString() {
61. *// TODO Auto-generated method stub*
62. return "Person [id=" + id + ", name=" + name + ", sex=" + sex + ", age=" + age + "]";
63. }
65. }

FKSqlSessionFactory.java

1. package com.factory;
3. import java.io.InputStream;
4. import org.apache.ibatis.io.Resources;
5. import org.apache.ibatis.session.SqlSession;
6. import org.apache.ibatis.session.SqlSessionFactory;
7. import org.apache.ibatis.session.SqlSessionFactoryBuilder;
9. public class FKSqlSessionFactory {
11. private static SqlSessionFactory sqlSessionFactory = null;
13. *// 初始化创建SqlSessionFactory对象*
14. static{
15. try {
16. *// 读取mybatis-config.xml文件*
17. InputStream inputStream = Resources.getResourceAsStream("mybatis-config.xml");
18. sqlSessionFactory = new SqlSessionFactoryBuilder()
19. .build(inputStream);
20. } catch (Exception e) {
21. e.printStackTrace();
22. }
23. }
25. *// 获取SqlSession对象的静态方法*
26. public static SqlSession getSqlSession(){
27. return sqlSessionFactory.openSession();
28. }
30. *// 获取SqlSessionFactory的静态方法*
31. public static SqlSessionFactory getSqlSessionFactory() {
32. return sqlSessionFactory;
33. }
35. }

CardMapper.xml

1. package com.mapper;
3. import org.apache.ibatis.annotations.Select;
5. import com.bean.Card;
7. public interface CardMapper {
8. @Select("select \* from tb\_card where id = #{id} ")
9. Card selectCardById(Integer id);
10. }

PersonMapper.xml

1. package com.mapper;
3. import org.apache.ibatis.annotations.One;
4. import org.apache.ibatis.annotations.Result;
5. import org.apache.ibatis.annotations.Results;
6. import org.apache.ibatis.annotations.Select;
7. import org.apache.ibatis.mapping.FetchType;
9. import com.bean.Person;
11. public interface PersonMapper {
12. @Select("select \* from tb\_person where id = #{id}")
13. @Results({
14. @Result(id=true,column="id",property="id"),
15. @Result(column="name",property="name"),
16. @Result(column="sex",property="sex"),
17. @Result(column="age",property="age"),
18. @Result(column="card\_id",property="card",one=@One(
19. select="com.mapper.CardMapper.selectCardById",fetchType=FetchType.EAGER))
20. })
21. Person selectPersonById(Integer id);
22. }

OneToOneTest.java

1. package com.test;
3. import org.apache.ibatis.session.SqlSession;
5. import com.bean.Person;
6. import com.factory.FKSqlSessionFactory;
7. import com.mapper.PersonMapper;
9. public class OneToOneTest {
11. public static void main(String[] args) {
12. *// TODO Auto-generated method stub*
13. SqlSession session=FKSqlSessionFactory.getSqlSession();
14. *// 获取PersonMapper实例*
15. PersonMapper pm = session.getMapper(PersonMapper.class);
16. *// 根据id查询Person对象，同时需要获得关联的Card对象*
17. Person p = pm.selectPersonById(1);
18. System.out.println(p);
19. *// 查看查询到的关联的Card对象*
20. System.out.println(p.getCard());
21. session.commit();
22. session.close();
23. }
25. }

截图：

版权声明：本文为博主原创文章，未经博主允许不得转载。 https://blog.csdn.net/linhaiyun\_ytdx/article/details/6894