# MyBatis中的动态SQL

## if 判断条件是否成立

|  |
| --- |
| <**select id="findAll" parameterType="User" resultType="User"**>  SELECT *\** FROM user where 1=1  <**if test="username!=null"**>  and username=#{username}  </**if**>  <**if test="password!=null"**>  and password=#{password}  </**if**> </**select**> |
| @Test  **public void** testFindAll2() {  **try** { *// 创建SqlSessionFactory* SqlSessionFactory sqlSessionFactory = **new** SqlSessionFactoryBuilder().build(Resources.*getResourceAsReader*(**"mybatis.xml"**)); *// 得到session对象* SqlSession session = sqlSessionFactory.openSession();  *//得到mapper接口的实现对象* UserMapper mapper = session.getMapper(UserMapper.**class**); *// 通过session对象操作数据库* User user = **new** User();  user.setUsername(**"admin"**);  List<User> list = mapper.findAll(user); *// 控制台打印list对象* System.***out***.println(list); *// 关闭session* session.close();  } **catch** (IOException e) {  e.printStackTrace();  }  } |
| ==> Preparing: SELECT \* FROM user where 1=1 and username=?  ==> Parameters: admin(String)  <== Columns: id, password, username  <== Row: 1, 123456, admin  <== Total: 1  [User{id=1, username='admin', password='123456'}] |

## choose 相当于 if elseif else 只有一个when或者otherwise被执行

|  |
| --- |
| <**select id="findAll" parameterType="User" resultType="User"**>  SELECT *\** FROM user where 1=1  <**choose**>  <**when test="username!=null"**>  and username=#{username}  </**when**>  <**when test="password!=null"**>  and password=#{password}  </**when**>  <**otherwise**>  and 2=2  </**otherwise**>  </**choose**> </**select**> |
| ==> Preparing: SELECT \* FROM user where 1=1 and username=?  ==> Parameters: admin(String)  <== Columns: id, password, username  <== Row: 1, 123456, admin  <== Total: 1  [User{id=1, username='admin', password='123456'}] |

## where 增加where条件，如果没有条件成立，where不会出现在sql语句中，反之出现sql语句中，并且忽略where后面的第一个and或者or

|  |
| --- |
| <**select id="findAll" parameterType="User" resultType="User"**>  SELECT *\** FROM user  <**where**>  <**if test="username!=null"**>  and username=#{username}  </**if**>  <**if test="password!=null"**>  and password=#{password}  </**if**>  </**where**> </**select**> |
| ==> Preparing: SELECT \* FROM user WHERE username=?  ==> Parameters: admin(String)  <== Columns: id, password, username  <== Row: 1, 123456, admin  <== Total: 1  [User{id=1, username='admin', password='123456'}] |

## trim 例如<trim prefix="where" prefixOverrides="and | or"> 忽略where后面的第一个and或者or，当然trim中所有的条件不成立，就不会出现where

|  |
| --- |
| <**select id="findAll" parameterType="User" resultType="User"**>  SELECT *\** FROM user  <**trim prefix="where" prefixOverrides="and | or"**>  <**if test="username!=null"**>  and username=#{username}  </**if**>  <**if test="password!=null"**>  and password=#{password}  </**if**>  </**trim**> </**select**> |
| ==> Preparing: SELECT \* FROM user where username=?  ==> Parameters: admin(String)  <== Columns: id, password, username  <== Row: 1, 123456, admin  <== Total: 1  [User{id=1, username='admin', password='123456'}] |

## set 后面必须满足于一个条件，不然不满足sql语法

set用于更新，增加set，如果没有条件成立，set不会出现在sql语句中，反之出现sql语句中，set忽略更新最后的一个字段语句后面的逗号

|  |
| --- |
| <**update id="update" parameterType="user"**>  update user  <**set**>  <**if test="username!=null"**>  username=#{username},  </**if**>  <**if test="password!=null"**>  password=#{password},  </**if**>  </**set**>  where id=#{id} </**update**> |
| @Test  **public void** testUpdate2() {  **try** { *// 创建SqlSessionFactory* SqlSessionFactory sqlSessionFactory = **new** SqlSessionFactoryBuilder().build(Resources.*getResourceAsReader*(**"mybatis.xml"**)); *// 得到session对象* SqlSession session = sqlSessionFactory.openSession();  *//得到mapper接口的实现对象* UserMapper mapper = session.getMapper(UserMapper.**class**); *// 通过session对象操作数据库* User user = **new** User();  user.setId(7);  user.setUsername(**"liubei"**);  mapper.update(user);  *//提交事务* session.commit(); *// 关闭session* session.close();  } **catch** (IOException e) {  e.printStackTrace();  }  } |
| ==> Preparing: update user SET username=? where id=?  ==> Parameters: liubei(String), 7(Integer)  <== Updates: 1 |

## foreach

### 单参list

|  |
| --- |
| <**select id="findAll2" resultType="User"**>  SELECT *\** FROM user where id in  <**foreach collection="list" item="item" open="(" close=")" separator=","**>  #{item}  </**foreach**> </**select**> |
| List<User> findAll2(List<Integer> list1); |
| @Test  **public void** testFindAll22() {  **try** { *// 创建SqlSessionFactory* SqlSessionFactory sqlSessionFactory = **new** SqlSessionFactoryBuilder().build(Resources.*getResourceAsReader*(**"mybatis.xml"**)); *// 得到session对象* SqlSession session = sqlSessionFactory.openSession();  *//得到mapper接口的实现对象* UserMapper mapper = session.getMapper(UserMapper.**class**); *// 通过session对象数据库* List<Integer> list1 = Arrays.*asList*(1, 3, 6);  List<User> list = mapper.findAll2(list1); *// 控制台打印list对象* System.***out***.println(list); *// 关闭session* session.close();  } **catch** (IOException e) {  e.printStackTrace();  }  } |
| ==> Preparing: SELECT \* FROM user where id in ( ? , ? , ? )  ==> Parameters: 1(Integer), 3(Integer), 6(Integer)  <== Columns: id, password, username  <== Row: 1, 123456, admin  <== Row: 3, 1111, admin1  <== Row: 6, 2333322, 3222  <== Total: 3  [User{id=1, username='admin', password='123456'}, User{id=3, username='admin1', password='1111'}, User{id=6, username='3222', password='2333322'}] |

### 单参数组

|  |
| --- |
| <**select id="findAll2" resultType="User"**>  SELECT *\** FROM user where id in  <**foreach collection="array" item="item" open="(" close=")" separator=","**>  #{item}  </**foreach**> </**select**> |
| List<User> findAll2(Integer[] array); |
| @Test  **public void** testFindAll22() {  **try** { *// 创建SqlSessionFactory* SqlSessionFactory sqlSessionFactory = **new** SqlSessionFactoryBuilder().build(Resources.*getResourceAsReader*(**"mybatis.xml"**)); *// 得到session对象* SqlSession session = sqlSessionFactory.openSession();  *//得到mapper接口的实现对象* UserMapper mapper = session.getMapper(UserMapper.**class**); *// 通过session对象数据库* Integer[] array=**new** Integer[]{1,3,6};  List<User> list = mapper.findAll2(array); *// 控制台打印list对象* System.***out***.println(list); *// 关闭session* session.close();  } **catch** (IOException e) {  e.printStackTrace();  }  } |
| ==> Preparing: SELECT \* FROM user where id in ( ? , ? , ? )  ==> Parameters: 1(Integer), 3(Integer), 6(Integer)  <== Columns: id, password, username  <== Row: 1, 123456, admin  <== Row: 3, 1111, admin1  <== Row: 6, 2333322, 3222  <== Total: 3  [User{id=1, username='admin', password='123456'}, User{id=3, username='admin1', password='1111'}, User{id=6, username='3222', password='2333322'}] |

### 多个参数使用map

|  |
| --- |
| <**select id="findAll2" resultType="User"**>  SELECT *\** FROM user where username like concat('%',#{username},'%') and id in  <**foreach collection="ids" item="item" open="(" close=")" separator=","**>  #{item}  </**foreach**> </**select**> |
| List<User> findAll2(Map<String,Object> map); |
| @Test  **public void** testFindAll22() {  **try** { *// 创建SqlSessionFactory* SqlSessionFactory sqlSessionFactory = **new** SqlSessionFactoryBuilder().build(Resources.*getResourceAsReader*(**"mybatis.xml"**)); *// 得到session对象* SqlSession session = sqlSessionFactory.openSession();  *//得到mapper接口的实现对象* UserMapper mapper = session.getMapper(UserMapper.**class**); *// 通过session对象数据库* Map<String,Object> map=**new** HashMap<>();  map.put(**"username"**, **"d"**);  map.put(**"ids"**, Arrays.*asList*(1,3,6));  List<User> list = mapper.findAll2(map); *// 控制台打印list对象* System.***out***.println(list); *// 关闭session* session.close();  } **catch** (IOException e) {  e.printStackTrace();  }  } |
| ==> Preparing: SELECT \* FROM user where username like '%d%' and id in ( ? , ? , ? )  ==> Parameters: 1(Integer), 3(Integer), 6(Integer)  <== Columns: id, password, username  <== Row: 1, 123456, admin  <== Row: 3, 1111, admin1  <== Total: 2  [User{id=1, username='admin', password='123456'}, User{id=3, username='admin1', password='1111'}] |

# Mybatis多表查询

## 一对一

一个对一个,人和身份证属于一对一,一个类只有一个另个类的引用

### 建立数据库表

|  |
| --- |
| CREATE TABLE `person` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `name` varchar(255) DEFAULT NULL,  PRIMARY KEY (`id`)  ) |
| CREATE TABLE `idcard` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `code` varchar(255) DEFAULT NULL,  `pid` int(11) DEFAULT NULL,  PRIMARY KEY (`id`)  ) |

### 实体类

|  |
| --- |
| **package** com.tjetc.entity;  **public class** Person {  **private int id**;  **private** String **name**;  **private** IdCard **idCard**;   **public** IdCard getIdCard() {  **return idCard**;  }   **public void** setIdCard(IdCard idCard) {  **this**.**idCard** = idCard;  }   **public** Person(String name) {  **this**.**name** = name;  }   **public** Person() {  }   **public** Person(**int** id, String name) {  **this**.**id** = id;  **this**.**name** = name;  }   @Override  **public** String toString() {  **return "Person{"** +  **"id="** + **id** +  **", name='"** + **name** + **'\''** +  **'}'**;  }   **public int** getId() {  **return id**;  }   **public void** setId(**int** id) {  **this**.**id** = id;  }   **public** String getName() {  **return name**;  }   **public void** setName(String name) {  **this**.**name** = name;  } } |
| **package** com.tjetc.entity;  **public class** IdCard {  **private int id**;  **private** String **code**;  **private** Person **person**;   @Override  **public** String toString() {  **return "IdCard{"** +  **"id="** + **id** +  **", code='"** + **code** + **'\''** +  **'}'**;  }   **public int** getId() {  **return id**;  }   **public void** setId(**int** id) {  **this**.**id** = id;  }   **public** String getCode() {  **return code**;  }   **public void** setCode(String code) {  **this**.**code** = code;  }   **public** Person getPerson() {  **return person**;  }   **public void** setPerson(Person person) {  **this**.**person** = person;  } } |

### 建立映射文件和映射接口

|  |
| --- |
| *PersonMapper.xml:*  *<?***xml version="1.0" encoding="utf-8"** *?>* **<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd" *>*** <**mapper namespace="com.tjetc.mapper.PersonMapper"**>  <**insert id="add" parameterType="person" useGeneratedKeys="true" keyProperty="id"**>  insert into person(name) values(#{name})  </**insert**> </**mapper**> |
| **package** com.tjetc.mapper;  **import** com.tjetc.entity.Person;  **public interface** PersonMapper {  **void** add(Person person); } |
| IdCardMapper.xml:  *<?***xml version="1.0" encoding="utf-8"** *?>* **<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd" *>*** <**mapper namespace="com.tjetc.mapper.IdCardMapper"**>  <**insert id="add" parameterType="idCard"**>  insert into idcard(code,pid) values (#{code},#{person.id})  </**insert**> </**mapper**> |
| **package** com.tjetc.mapper;  **import** com.tjetc.entity.IdCard;  **public interface** IdCardMapper {  **void** add(IdCard idCard); } |

### mybatis.xml

|  |
| --- |
| <**mappers**>  <**package name="com.tjetc.mapper"**/> *<!-- <mapper resource="com/tjetc/mapper/UserMapper.xml"></mapper>-->* </**mappers**> |

### 测试方法

|  |
| --- |
| @Test  **public void** testAdd() { *// 创建SqlSessionFactory* SqlSessionFactory sqlSessionFactory = **null**;  **try** {  sqlSessionFactory = **new** SqlSessionFactoryBuilder().build(Resources.*getResourceAsReader*(**"mybatis.xml"**)); *// 得到session对象* SqlSession session = sqlSessionFactory.openSession();  *//得到mapper* PersonMapper personMapper = session.getMapper(PersonMapper.**class**);  *//向数据库person表添加一条记录* Person person = **new** Person(**"张三"**);  personMapper.add(person);  System.***out***.println(person);  *//创建idCard对象* IdCard idCard = **new** IdCard();  idCard.setCode(**"12345678234567"**);  idCard.setPerson(person);  *//插入数据库* IdCardMapper cardMapper = session.getMapper(IdCardMapper.**class**);  cardMapper.add(idCard);   session.commit();  session.close();  } **catch** (IOException e) {  e.printStackTrace();  }  } |

### 运行效果

|  |
| --- |
| ==> Preparing: insert into person(name) values(?)  ==> Parameters: 张三(String)  <== Updates: 1  Person{id=1, name='张三'}  ==> Preparing: insert into idcard(code,pid) values (?,?)  ==> Parameters: 12345678234567(String), 1(Integer)  <== Updates: 1 |
|  |

### 课堂练习

使用mybatis实现person和idcard表一对一的插入数据库表

### resultMap

resultMap:结果映射,查询结果列与实体类对象的属性之间进行映射,还能进行关联的映射.

### 一对一的关联嵌套select

|  |
| --- |
| PersonMapper.xml:  *<?***xml version="1.0" encoding="utf-8"** *?>* **<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd" *>*** <**mapper namespace="com.tjetc.mapper.PersonMapper"**>  <**insert id="add" parameterType="com.tjetc.entity.Person" useGeneratedKeys="true" keyProperty="id"**>  insert into person(name) values(#{name})  </**insert**>  <**select id="findById" parameterType="int" resultMap="personMap"**>  select *\** from person where id=#{id}  </**select**>  <**resultMap id="personMap" type="com.tjetc.entity.Person"**>  <**id column="id" property="id"**></**id**>  <**result column="name" property="name"**></**result**>  <**association property="idCard" column="id" select="findByPid"**></**association**>  </**resultMap**>  <**select id="findByPid" parameterType="int" resultType="com.tjetc.entity.IdCard"**>  select *\** from idcard where pid=#{id}  </**select**> </**mapper**> |
| **public interface** PersonMapper {  **void** add(Person person);  Person findById(**int** id); } |
| @Test  **public void** testFindById() { *// 创建SqlSessionFactory* SqlSessionFactory sqlSessionFactory = **null**;  **try** {  sqlSessionFactory = **new** SqlSessionFactoryBuilder().build(Resources.*getResourceAsReader*(**"mybatis.xml"**)); *// 得到session对象* SqlSession session = sqlSessionFactory.openSession();  *//得到mapper* PersonMapper personMapper = session.getMapper(PersonMapper.**class**);  *//调用PersonMapper的方法* Person person = personMapper.findById(1);  System.***out***.println(person);  IdCard idCard = person.getIdCard();  System.***out***.println(idCard);  session.close();  } **catch** (IOException e) {  e.printStackTrace();  }  } |
| ==> Preparing: select \* from person where id=?  ==> Parameters: 1(Integer)  <== Columns: id, name  <== Row: 1, 张三  ====> Preparing: select \* from idcard where pid=?  ====> Parameters: 1(Integer)  <==== Columns: id, code, pid  <==== Row: 1, 12345678234567, 1  <==== Total: 1  <== Total: 1  Person{id=1, name='张三'}  IdCard{id=1, code='12345678234567'} |

### 课堂练习

一对一的关联嵌套select

### 一对一的关联嵌套resultMap

|  |
| --- |
| <**select id="findById" parameterType="int" resultMap="personMap"**>  select p.id,p.name,c.id cid,c.code from person p inner join idcard c on p.id=c.pid where p.id=#{id}  </**select**> <**resultMap id="personMap" type="com.tjetc.entity.Person"**>  <**id column="id" property="id"**></**id**>  <**result column="name" property="name"**></**result**>  <**association property="idCard" resultMap="cardMap"**></**association**> </**resultMap**> <**resultMap id="cardMap" type="IdCard"**>  <**id column="cid" property="id"**></**id**>  <**result column="code" property="code"**></**result**> </**resultMap**> |
| @Test  **public void** testFindById() { *// 创建SqlSessionFactory* SqlSessionFactory sqlSessionFactory = **null**;  **try** {  sqlSessionFactory = **new** SqlSessionFactoryBuilder().build(Resources.*getResourceAsReader*(**"mybatis.xml"**)); *// 得到session对象* SqlSession session = sqlSessionFactory.openSession();  *//得到mapper* PersonMapper personMapper = session.getMapper(PersonMapper.**class**);  *//调用PersonMapper的方法* Person person = personMapper.findById(1);  System.***out***.println(person);  IdCard idCard = person.getIdCard();  System.***out***.println(idCard);  session.close();  } **catch** (IOException e) {  e.printStackTrace();  }  } |
| ==> Preparing: select p.\*,c.id cid,c.code from person p,idcard c where p.id=c.pid and p.id=?  ==> Parameters: 1(Integer)  <== Columns: id, name, cid, code  <== Row: 1, 张三, 1, 12345678234567  <== Total: 1  Person{id=1, name='张三'}  IdCard{id=1, code='12345678234567'} |

## 一对多

### 建表

myclass

student

|  |
| --- |
| CREATE TABLE `myclass` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `name` varchar(255) DEFAULT NULL,  PRIMARY KEY (`id`)  ) |
| CREATE TABLE `student` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `name` varchar(255) DEFAULT NULL,  `cid` int(11) DEFAULT NULL,  PRIMARY KEY (`id`)  ) |

### 实体类

|  |
| --- |
| **package** com.tjetc.domain;  **import** java.util.ArrayList; **import** java.util.List;  **public class** MyClass {  **private int id**;  **private** String **name**;  **private** List<Student> **students**=**new** ArrayList<>();   **public** List<Student> getStudents() {  **return students**;  }   **public void** setStudents(List<Student> students) {  **this**.**students** = students;  }   **public** MyClass(String name) {  **this**.**name** = name;  }   **public** MyClass(**int** id, String name) {  **this**.**id** = id;  **this**.**name** = name;  }   **public** MyClass() {  }   @Override  **public** String toString() {  **return "MyClass{"** +  **"id="** + **id** +  **", name='"** + **name** + **'\''** +  **'}'**;  }   **public int** getId() {  **return id**;  }   **public void** setId(**int** id) {  **this**.**id** = id;  }   **public** String getName() {  **return name**;  }   **public void** setName(String name) {  **this**.**name** = name;  } } |
| **package** com.tjetc.entity;  **public class** Student {  **private int id**;  **private** String **name**;  **private** MyClass **myClass**;   **public** Student(String name) {  **this**.**name** = name;  }   **public** Student() {  }   @Override  **public** String toString() {  **return "Student{"** +  **"id="** + **id** +  **", name='"** + **name** + **'\''** +  **'}'**;  }   **public int** getId() {  **return id**;  }   **public void** setId(**int** id) {  **this**.**id** = id;  }   **public** String getName() {  **return name**;  }   **public void** setName(String name) {  **this**.**name** = name;  }   **public** MyClass getMyClass() {  **return myClass**;  }   **public void** setMyClass(MyClass myClass) {  **this**.**myClass** = myClass;  } } |

### 映射文件和映射接口

|  |
| --- |
| MyClassMapper.xml:  *<?***xml version="1.0" encoding="utf-8"** *?>* **<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd" *>*** <**mapper namespace="com.tjetc.mapper.MyClassMapper"**>  <**insert id="add" parameterType="MyClass" useGeneratedKeys="true" keyProperty="id"**>  insert into myclass(name) values(#{name})  </**insert**> </**mapper**> |
| **package** com.tjetc.mapper;  **import** com.tjetc.entity.MyClass;  **public interface** MyClassMapper {  **void** add(MyClass myClass); } |
| *StudentMapper.xml:*  *<?***xml version="1.0" encoding="utf-8"** *?>* **<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd" *>*** <**mapper namespace="com.tjetc.mapper.StudentMapper"**>  <**insert id="add" parameterType="Student"**>  insert into student(name,cid) values (#{name},#{myClass.id});  </**insert**> </**mapper**> |
| **package** com.tjetc.mapper;  **import** com.tjetc.entity.Student;  **public interface** StudentMapper {  **void** add(Student student); } |
| @Test  **public void** testAdd(){  **try** {  SqlSessionFactory factory = **new** SqlSessionFactoryBuilder().build(Resources.*getResourceAsReader*(**"mybatis.xml"**));  SqlSession session = factory.openSession();  MyClassMapper myClassMapper = session.getMapper(MyClassMapper.**class**);  StudentMapper studentMapper = session.getMapper(StudentMapper.**class**);  MyClass myClass = **new** MyClass(**"1班"**);  myClassMapper.add(myClass);  *// 创建3个学生* Student student1 = **new** Student(**"zs"**);  Student student2 = **new** Student(**"ls"**);  Student student3 = **new** Student(**"zl"**); *// 设置班级* student1.setMyClass(myClass);  student2.setMyClass(myClass);  student3.setMyClass(myClass);  *// 保存学生* studentMapper.add(student1);  studentMapper.add(student2);  studentMapper.add(student3); *// 提交事务* session.commit(); *// close* session.close();   } **catch** (IOException e) {  e.printStackTrace();  }  } |
| ==> Preparing: insert into myclass(name) values(?)  ==> Parameters: 1班(String)  <== Updates: 1  ==> Preparing: insert into student(name,cid) values (?,?);  ==> Parameters: zs(String), 1(Integer)  <== Updates: 1  ==> Preparing: insert into student(name,cid) values (?,?);  ==> Parameters: ls(String), 1(Integer)  <== Updates: 1  ==> Preparing: insert into student(name,cid) values (?,?);  ==> Parameters: zl(String), 1(Integer)  <== Updates: 1 |
|  |

### 一对多的集合嵌套select collection

|  |
| --- |
| MyClassMapper.xml:  <**select id="findById" parameterType="int" resultMap="classMap"**>  select *\** from myclass where id=#{id} </**select**> <**resultMap id="classMap" type="MyClass"**>  <**id column="id" property="id"**></**id**>  <**result column="name" property="name"**></**result**>  <**collection property="students" ofType="Student" column="id" select="findByCid"**></**collection**> </**resultMap**> <**select id="findByCid" parameterType="int" resultType="Student"**>  select *\** from student where cid=#{cid} </**select**> |
| MyClass findById(**int** id); |
| @Test **public void** testFind() {  **try** {  SqlSessionFactory factory = **new** SqlSessionFactoryBuilder().build(Resources.*getResourceAsReader*(**"mybatis.xml"**));  SqlSession session = factory.openSession();  MyClassMapper myClassMapper = session.getMapper(MyClassMapper.**class**);  MyClass myClass = myClassMapper.findById(1);  System.***out***.println(myClass);  List<Student> students = myClass.getStudents();  **for** (Student student : students) {  System.***out***.println(student);  }  session.close();  } **catch** (IOException e) {  e.printStackTrace();  } } |
| ==> Preparing: select \* from myclass where id=?  ==> Parameters: 1(Integer)  <== Columns: id, name  <== Row: 1, 1班  <== Total: 1  ==> Preparing: select \* from student where cid=?  ==> Parameters: 1(Integer)  <== Columns: id, name, cid  <== Row: 1, zs, 1  <== Row: 2, ls, 1  <== Row: 3, zl, 1  <== Total: 3  MyClass{id=1, name='1班'}  Student{id=1, name='zs'}  Student{id=2, name='ls'}  Student{id=3, name='zl'} |

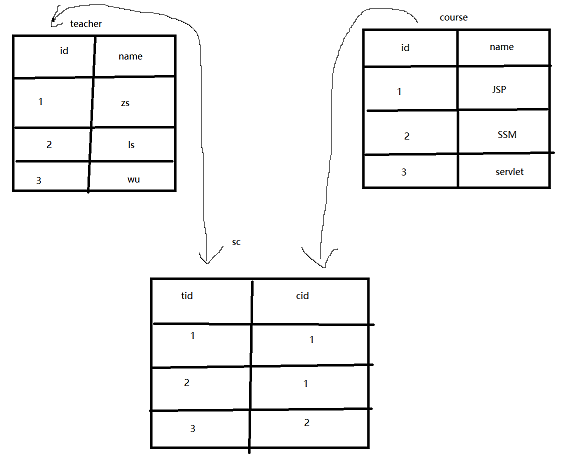
### 一对多的集合嵌套resultMap collection

|  |
| --- |
| <**select id="findById" parameterType="int" resultMap="classMap"**>  SELECT c.*\**,s.id sid,s.name sname FROM myclass c,student s where c.id=s.cid and c.id=#{id} </**select**> <**resultMap id="classMap" type="MyClass"**>  <**id column="id" property="id"**></**id**>  <**result column="name" property="name"**></**result**>  <**collection property="students" ofType="Student" resultMap="studentMap"**></**collection**> </**resultMap**> <**resultMap id="studentMap" type="Student"**>  <**id column="sid" property="id"**></**id**>  <**result column="sname" property="name"**></**result**> </**resultMap**> |
| ==> Preparing: SELECT c.\*,s.id sid,s.name sname FROM myclass c,student s where c.id=s.cid and c.id=?  ==> Parameters: 1(Integer)  <== Columns: id, name, sid, sname  <== Row: 1, 1班, 1, zs  <== Row: 1, 1班, 2, ls  <== Row: 1, 1班, 3, zl  <== Total: 3  MyClass{id=1, name='1班'}  Student{id=1, name='zs'}  Student{id=2, name='ls'}  Student{id=3, name='zl'} |

## 多对多

### 建表

教师选课



|  |
| --- |
| CREATE TABLE `teacher` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `name` varchar(255) DEFAULT NULL,  PRIMARY KEY (`id`)  ); |
| CREATE TABLE `course` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `name` varchar(255) DEFAULT NULL,  PRIMARY KEY (`id`)  ) ; |
| CREATE TABLE `sc` (  `tid` int(11) NOT NULL,  `cid` int(11) DEFAULT NULL  ) ; |

### 实体类

|  |
| --- |
| **package** com.tjetc.entity;  **import** java.util.ArrayList; **import** java.util.List;  **public class** Teacher {  **private int id**;  **private** String **name**;  **private** List<Course> **courses** = **new** ArrayList<>();   **public** List<Course> getCourses() {  **return courses**;  }   **public void** setCourses(List<Course> courses) {  **this**.**courses** = courses;  }   **public** Teacher() {  }   **public** Teacher(String name) {  **this**.**name** = name;  }   @Override  **public** String toString() {  **return "Teacher{"** +  **"id="** + **id** +  **", name='"** + **name** + **'\''** +  **'}'**;  }   **public int** getId() {  **return id**;  }   **public void** setId(**int** id) {  **this**.**id** = id;  }   **public** String getName() {  **return name**;  }   **public void** setName(String name) {  **this**.**name** = name;  } } |
| **package** com.tjetc.entity;  **import** java.util.ArrayList; **import** java.util.List;  **public class** Course {  **private int id**;  **private** String **name**;  **private** List<Teacher> **teachers** =**new** ArrayList<>();   **public** List<Teacher> getTeachers() {  **return teachers**;  }   **public void** setTeachers(List<Teacher> teachers) {  **this**.**teachers** = teachers;  }   **public** Course() {  }   **public** Course(String name) {  **this**.**name** = name;  }   @Override  **public** String toString() {  **return "Course{"** +  **"id="** + **id** +  **", name='"** + **name** + **'\''** +  **'}'**;  }   **public int** getId() {  **return id**;  }   **public void** setId(**int** id) {  **this**.**id** = id;  }   **public** String getName() {  **return name**;  }   **public void** setName(String name) {  **this**.**name** = name;  } } |

### 映射文件和接口

|  |
| --- |
| TeacherMapper.xml:  *<?***xml version="1.0" encoding="utf-8"** *?>* **<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd" *>*** <**mapper namespace="com.tjetc.mapper.TeacherMapper"**>  <**insert id="add" parameterType="com.tjetc.entity.Teacher" useGeneratedKeys="true" keyProperty="id"**>  insert into teacher(name) values(#{name})  </**insert**>  <**insert id="addSc"**>  insert into sc(tid,cid) values(#{tid},#{cid})  </**insert**>  </**mapper**> |
| **package** com.tjetc.mapper;  **import** com.tjetc.entity.Teacher; **import** org.apache.ibatis.annotations.Param;  **public interface** TeacherMapper {  **void** add(Teacher teacher);   */\*\*  \* 添加sc表  \*  \** ***@param t\_id*** *tid  \** ***@param c\_id*** *cid  \*/* **void** addSc(@Param(**"tid"**) **int** t\_id, @Param(**"cid"**) **int** c\_id); } |
| CourseMapper.xml:  *<?***xml version="1.0" encoding="utf-8"** *?>* **<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd" *>*** <**mapper namespace="com.tjetc.mapper.CourseMapper"**>  <**insert id="add" parameterType="com.tjetc.entity.Course" useGeneratedKeys="true" keyProperty="id"**>  insert into course(name) values(#{name})  </**insert**> </**mapper**> |
| **public interface** CourseMapper {  **void** add(Course course); } |
| **package** com.tjetc;  **import** com.tjetc.common.SqlSessionFactoryUtil; **import** com.tjetc.domain.Course; **import** com.tjetc.domain.Teacher; **import** com.tjetc.mapper.CourseMapper; **import** com.tjetc.mapper.TeacherMapper; **import** org.apache.ibatis.session.SqlSession; **import** org.junit.Test;  **public class** TeacherAndCourseTest {  @Test  **public void** add() {  SqlSession sqlSession = SqlSessionFactoryUtil.*getSqlSession*(**"mybatis.xml"**);  TeacherMapper teacherMapper = sqlSession.getMapper(TeacherMapper.**class**);  CourseMapper courseMapper = sqlSession.getMapper(CourseMapper.**class**);   *//new三门课程* Course course1 = **new** Course(**"java"**);  Course course2 = **new** Course(**"JSP"**);  Course course3 = **new** Course(**"Mybatis"**);  *//三门课程添加到数据库* courseMapper.add(course1);  courseMapper.add(course2);  courseMapper.add(course3);   *//new 三个老师* Teacher teacher1 = **new** Teacher(**"zs"**);  Teacher teacher2 = **new** Teacher(**"ls"**);  Teacher teacher3 = **new** Teacher(**"zl"**);  *//三个老师添加到数据库* teacherMapper.add(teacher1);  teacherMapper.add(teacher2);  teacherMapper.add(teacher3);   *//zs老师选3门课程* teacherMapper.addSc(teacher1.getId(),course1.getId());  teacherMapper.addSc(teacher1.getId(),course2.getId());  teacherMapper.addSc(teacher1.getId(),course3.getId());   *//ls老师选1和2课程* teacherMapper.addSc(teacher2.getId(),course1.getId());  teacherMapper.addSc(teacher2.getId(),course2.getId());   *//wu老师选2和3课程* teacherMapper.addSc(teacher3.getId(),course2.getId());  teacherMapper.addSc(teacher3.getId(),course3.getId());   sqlSession.commit();   sqlSession.close();  } } |
| ==> Preparing: insert into course (name) values (?)  ==> Parameters: Java(String)  <== Updates: 1  ==> Preparing: insert into course (name) values (?)  ==> Parameters: JSP(String)  <== Updates: 1  ==> Preparing: insert into course (name) values (?)  ==> Parameters: MyBatis(String)  <== Updates: 1  ==> Preparing: insert into student (name) values (?)  ==> Parameters: zs(String)  <== Updates: 1  ==> Preparing: insert into student (name) values (?)  ==> Parameters: ls(String)  <== Updates: 1  ==> Preparing: insert into student (name) values (?)  ==> Parameters: zl(String)  <== Updates: 1  ==> Preparing: insert into sc (sid,cid) values (?,?)  ==> Parameters: 1(Integer), 1(Integer)  <== Updates: 1  ==> Preparing: insert into sc (sid,cid) values (?,?)  ==> Parameters: 1(Integer), 2(Integer)  <== Updates: 1  ==> Preparing: insert into sc (sid,cid) values (?,?)  ==> Parameters: 1(Integer), 3(Integer)  <== Updates: 1  ==> Preparing: insert into sc (sid,cid) values (?,?)  ==> Parameters: 2(Integer), 2(Integer)  <== Updates: 1  ==> Preparing: insert into sc (sid,cid) values (?,?)  ==> Parameters: 2(Integer), 3(Integer)  <== Updates: 1  ==> Preparing: insert into sc (sid,cid) values (?,?)  ==> Parameters: 3(Integer), 1(Integer)  <== Updates: 1  ==> Preparing: insert into sc (sid,cid) values (?,?)  ==> Parameters: 3(Integer), 3(Integer)  <== Updates: 1 |

### 多对多的集合嵌套select collection

|  |
| --- |
| <**select id="findById" parameterType="int" resultMap="teacherMap"**>  select *\** from teacher where id=#{id} </**select**> <**resultMap id="teacherMap" type="com.tjetc.entity.Teacher"**>  <**id column="id" property="id"**></**id**>  <**result column="name" property="name"**></**result**>  <**collection property="courses" ofType="com.tjetc.entity.Course" column="id" select="findByTid"**></**collection**> </**resultMap**> <**select id="findByTid" parameterType="int" resultType="com.tjetc.entity.Course"**>  select c.\* from course c  inner join sc on c.id = sc.cid  where sc.tid=#{id} </**select**> |
| @Test **public void** testFind() {  **try** {  SqlSessionFactory factory = **new** SqlSessionFactoryBuilder().build(Resources.*getResourceAsReader*(**"mybatis.xml"**));  SqlSession session = factory.openSession();  TeacherMapper teacherMapper = sqlSession.getMapper(TeacherMapper.**class**); Teacher teacher = teacherMapper.findById(1); System.***out***.println(teacher); *//通过teacher对象获取course对象* List<Course> courses = teacher.getCourses(); System.***out***.println(courses); sqlSession.close();  } **catch** (IOException e) {  e.printStackTrace();  } } |
| ==> Preparing: select \* from teacher where id=?  ==> Parameters: 1(Integer)  <== Columns: id, name  <== Row: 1, zs  ====> Preparing: select c.\* from course c inner join sc on c.id = sc.cid where sc.tid=?  ====> Parameters: 1(Integer)  <==== Columns: id, name  <==== Row: 4, java  <==== Row: 5, JSP  <==== Row: 6, Mybatis  <==== Total: 3  <== Total: 1  Teacher{id=1, name='zs'}  [Course{id=4, name='java'}, Course{id=5, name='JSP'}, Course{id=6, name='Mybatis'}]  Resetting autocommit to true on JDBC Connection [com.mysql.jdbc.JDBC4Connection@3911c2a7]  Closing JDBC Connection [com.mysql.jdbc.JDBC4Connection@3911c2a7]  Returned connection 957465255 to pool.  Process finished with exit code 0 |

### 多对多的集合嵌套resultMap collection

|  |
| --- |
| <**select id="findById2" parameterType="int" resultMap="teacherMap2"**>  select t.id tid,t.`name` tname,c.id cid,c.name cname from teacher t  left join sc on t.id = sc.tid  left join course c on sc.cid = c.id  where t.id=#{id} </**select**> <**resultMap id="teacherMap2" type="com.tjetc.entity.Teacher"**>  <**id column="tid" property="id"**></**id**>  <**result column="tname" property="name"**></**result**>  <**collection property="courses" ofType="com.tjetc.entity.Course" resultMap="courseMap"**></**collection**> </**resultMap**> <**resultMap id="courseMap" type="com.tjetc.entity.Course"**>  <**id column="cid" property="id"**></**id**>  <**result column="cname" property="name"**></**result**> </**resultMap**> |
| @Test **public void** testFind() {  **try** {  SqlSessionFactory factory = **new** SqlSessionFactoryBuilder().build(Resources.*getResourceAsReader*(**"mybatis.xml"**));  SqlSession session = factory.openSession();  TeacherMapper teacherMapper = sqlSession.getMapper(TeacherMapper.**class**); Teacher teacher = teacherMapper.findById2(1); System.***out***.println(teacher); *//通过teacher对象获取course对象* List<Course> courses = teacher.getCourses(); System.***out***.println(courses); sqlSession.close();  } **catch** (IOException e) {  e.printStackTrace();  } } |
| ==> Preparing: select t.id tid,t.`name` tname,c.id cid,c.name cname from teacher t inner join sc on t.id = sc.tid inner join course c on sc.cid = c.id where t.id=?  ==> Parameters: 1(Integer)  <== Columns: tid, tname, cid, cname  <== Row: 1, zs, 4, java  <== Row: 1, zs, 5, JSP  <== Row: 1, zs, 6, Mybatis  <== Total: 3  Teacher{id=1, name='zs'}  [Course{id=4, name='java'}, Course{id=5, name='JSP'}, Course{id=6, name='Mybatis'}]  Resetting autocommit to true on JDBC Connection [com.mysql.jdbc.JDBC4Connection@3911c2a7]  Closing JDBC Connection [com.mysql.jdbc.JDBC4Connection@3911c2a7]  Returned connection 957465255 to pool.  Process finished with exit code 0 |