课程目标

1. 实现mybatis单表的增删改查(重点)
2. 利用ResultMap标签定义返回结果
3. 省略Dao层实现类
4. 实现mybatis单表的增删改查
5. 实现添加功能
   1. 定义添加持久化操作

|  |
| --- |
| *<!--添加持久化操作 建议:id的属性值与方法名相同  parameterType持定义参数的类型  在sql语句使用#{属性名|键名}进行占位 -->* <**insert id="addGrade" parameterType="com.team.project.entity.Grade"**>  insert into grade(gid,gname) values(#{gid},#{gname}) </**insert**> |

1. 实现修改功能
   1. 定义修改的持久化操作

|  |
| --- |
| *<!--修改持久化操作-->* <**update id="updateGrade" parameterType="com.team.project.entity.Grade"**>  update grade set gname=#{gname} where gid=#{gid} </**update**> |

1. 实现删除功能
   1. 定义删除的持久化操作

|  |
| --- |
| *<!-- 删除持久化 parameterType 指定参数类型 注意:mybatis定义内置的java类型别名 java类型 别名 java.lang.Integer int -->* <**delete id="deleteGrade" parameterType="int"**>  delete from grade where gid=#{gid} </**delete**> |

1. 实现查询单条记录

|  |
| --- |
| *<!--定义查询单条-->* <**select id="getGrade" parameterType="int" resultType="com.team.project.entity.Grade"**>  select *\** from grade where gid=#{gid} </**select**> |

1. 查询

|  |
| --- |
| <**select id="getAllGrade" resultType="com.team.project.entity.Grade"**>  select *\** from grade </**select**> |

1. sqlSession调用持久化操作的方法

1.添加方法

sqlSession.insert(“添加持久化操作”[,参数对象])

2.修改方法

sqlSession.update(“修改持久化操作”[,参数对象])

3.删除方法

sqlSession.delete(“删除持久化操作”[,参数对象])

4.查询单条

sqlSession.selectOne(“查询的持久化操作” [,参数对象]);

5.查询

sqlSession.selectList(“查询的持久化操作” [,参数对象]);

注意: *如果执行添加，修改，删除操作时，mybatis需要基于事务执行(需要提交或者回滚)*

三、（了解）ResultMap标签自定义返回结果类型(用在返回复杂结构)

示例:执行查询时使用ResultMap属性返回结果

注意:查询持久化操作resultType和resultMap两个属性不能同时存在,推荐使用resultType属性

resultType指定返回实体的属性必需要与表的字段名称相同才返回获取结果

resultMap指定自定义返回结果(指定属性列的对应关联)

|  |
| --- |
| *<!--ResultMap自定义结果类型-->* <**resultMap id="gradeResult" type="com.team.project.entity.Grade"**>  *<!--指定哪个属性与哪个列对应-->* <**id property="gid" column="gid"**></**id**>  *<!--<result property="gid" column="gid"></result>-->* <**result property="gname" column="gname"**></**result**>  </**resultMap**>  <**select id="getAllGrade" resultMap="gradeResult"**>  select gid,gname from grade </**select**> |

1. (重点)使用sqlSession对象的getMapper方法动态生成代理实现类(省dao层实现类)

在mybatis开发中:无需编写dao层实现类

*注意:sql映射文件中的命名空间必需是接口的限定名称, 持久化操作的id属性值必需和接口的方法名称相同*

|  |
| --- |
| **public static void** main(String[] args) {  *//查询所有年级进行显示输出到控制台  //利用sqlSession的getMapper方法生成接口的动态代理实现类* **try** {  SqlSession sqlSession=MyBatisUtil.*getSession*(); *//获取sqlSession*  **//使用getMapper方法生成接口动态代理实现类  //必需满足以下条件:  // 1.sql映射文件的命名空间的值必需是接口的限定名  // 2.接口中的方法名称必需与持久化操作id值相同  GradeDao gradeDao=sqlSession.getMapper(GradeDao.class);**   List<Grade> list=gradeDao.getAllGrade();  *//显示* list.forEach(grade -> {  System.***out***.println(**"编号:"**+grade.getGid()+**"名称"**+grade.getGname());  });  } **catch** (Exception e) {  e.printStackTrace();  }**finally** {  MyBatisUtil.*closeSession*(); *//关闭sqlSession* } } |

MyBatisUtil工具类，关闭和获取sqlSession

五、其它:mybatis定义内置的java类型别名

内置类型别名 java类型

\_int int

\_short short

\_long long

\_float float

int java.lang.Integer

float java.lang.Float

double java.lang.Double

list java.uitl.List

arraylist java.uitl.ArrayList

map java.uitl.Map