课程目标

1. Mybatis指定实体类型的别名 (会)
2. Mybatis反向工程(重点，熟练)
3. 使用反向工程selectByExample方法实现查询(会)
4. Mybatis连接查询（下节课继续讲）
5. 配置mybatis的实体类型的别名 (修改:mybatis的配置文件)

|  |
| --- |
| <**typeAliases**>  *<!--1.利用typeAlias为某个实体类取别名  <typeAlias type="com.kgc.entity.Grade" alias="Grade"></typeAlias>  -->  <!--2.使用package为包下所有的实体类型取别名  name 指定包名、生成实体的别名就是类名  -->* <**package name="com.team.project.entity"**></**package**> </**typeAliases**> |

注意:

Configuration标签出现子元素的顺序:

properties,settings,typeAliases,typeHandlers,objectFactory,objectWrapperFactory,plugins,environments,databaseIdProvider,mappers

1. (重点)使用Mybatis的反向工程(自动生成:实体类、接口、sql映射文件)

回顾:mybatis的dao层组件有哪些:实体、接口、sql映射文件

1. **实现mybatis的反向工程的步骤** 
   1. **在pom文件中导入mybatis自动生成的jar包(mybatis-generator-core)**

|  |
| --- |
| *<!--mybatis自动生成的依赖包-->* <**dependency**>  <**groupId**>org.mybatis.generator</**groupId**>  <**artifactId**>mybatis-generator-core</**artifactId**>  <**version**>1.4.1</**version**> </**dependency**> |

**1.2 在工程中编写mybatis自动生成的配置文件(generator.xml)**

**注意:放在resources目录中**

|  |
| --- |
| *<?***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**>  *<!—1.指定数据的驱动包-->* <**classPathEntry location="D:\mysql-connector-java-5.1.0-bin.jar"** />   <**context id="Mysql2Tables" targetRuntime="MyBatis3"**>  <**commentGenerator**>  <**property name="suppressDate" value="true"**></**property**>  <**property name="suppressAllComments" value="true"**></**property**>  </**commentGenerator**>   <**jdbcConnection driverClass="com.mysql.jdbc.Driver"  connectionURL="jdbc:mysql://localhost:3306/k9051db"  userId="root"  password="123"**>  </**jdbcConnection**>   <**javaTypeResolver** >  <**property name="forceBigDecimals" value="false"** />  </**javaTypeResolver**>   *<!--指定生成实体所有在的包*  *注意:*  *如果项目是创建的工程* **targetProject=”src/main/java”**  **如果项目是创建的模块 targetProject=”模块项目名称/ src/main/java”**  *-->* <**javaModelGenerator targetPackage="com.kgc.grade.entity" targetProject="mybatis\_ch04/src/main/java"**>  <**property name="enableSubPackages" value="true"** />  <**property name="trimStrings" value="true"** />  </**javaModelGenerator**>   *<!--指定sql映射文件的位置-->* <**sqlMapGenerator targetPackage="com.kgc.grade.dao" targetProject="mybatis\_ch04/src/main/java"**>  <**property name="enableSubPackages" value="true"** />  </**sqlMapGenerator**>   *<!--指定接口存放的位置-->* <**javaClientGenerator type="XMLMAPPER" targetPackage="com.kgc.grade.dao" targetProject="mybatis\_ch04/src/main/java"**>  <**property name="enableSubPackages" value="true"** />  </**javaClientGenerator**>   *<!--添加需要反向工程的表-->* <**table tableName="grade" enableCountByExample="false" enableUpdateByExample="false"  enableDeleteByExample="false" enableSelectByExample="false" selectByExampleQueryId="false"**>  </**table**>  <**table tableName="students" enableCountByExample="false" enableUpdateByExample="false"  enableDeleteByExample="false" enableSelectByExample="false" selectByExampleQueryId="false"**>  </**table**>  </**context**> </**generatorConfiguration**> |

**1.3 运行自动生成的工具类(MybatisGeneratorUtil.java)**

将工具类复制到项目中，并运行即可

1.4 修改mybatis的配置文件，加载所有的sql映射文件

|  |
| --- |
| *<!-- 加载sql映射文件-->* <**mappers**>  *<!--添加映射文件  <mapper resource="com/kgc/mapper/GradeMapper.xml"></mapper>  <mapper resource="com/kgc/mapper/StudentMapper.xml"></mapper>  -->  <!--加载某个包下所有sql映射文件-->* <**package name="com.kgc.mapper"**></**package**> </**mappers**> |

1. 使用反向工程生成文件(代码)

>> 测试添加:添加年级

略

>> 测试查询所有

* + - 添加查询所有年级的方法

|  |
| --- |
| *//查询所有年级* List<Grade> getAllGrade(); |

* + - 添加查询所有年级的持久操作

|  |
| --- |
| *<!--查询所有年级-->* <**select id="getAllGrade" resultType="Grade"**>  select *\** from grade </**select**> |

* + - 测试查询所有

|  |
| --- |
| **public static void** main(String[] args) {  *//1.使用mybatis工具类获得sqlSession* SqlSession sqlSession=MyBatisUtil.*getSession*();  *//2.生成dao层的对象* GradeMapper gradeMapper=sqlSession.getMapper(GradeMapper.**class**);  *//3.调用dao层方法查询所有* List<Grade> list=gradeMapper.getAllGrade();  *//4.关闭资源* MyBatisUtil.*closeSession*();   *//显示结果* System.***out***.println(**"编号\t名称"**);  list.forEach(grade -> {  System.***out***.println(grade.getGid()+**"\t"**+grade.getGname());  }); } |

//使用mybatis的工具类获取sqlSession

1. 利用反向工程的selectByExample方法进行查询(条件查询)
2. 修改自动生成的配置文件，将以下开关打开，启动selectByExample

|  |
| --- |
| <**table tableName="grade" enableCountByExample="false" enableUpdateByExample="false"  enableDeleteByExample="false" enableSelectByExample="true" selectByExampleQueryId="false"**> </**table**> |

1. 调用dao层的selectByExample实现查询

|  |
| --- |
| **public static void** main(String[] args) {  *//获取sqlSession* SqlSession sqlSession=MyBatisUtil.*getSession*();  *//创建dao层对象* StudentMapper studentMapper=sqlSession.getMapper(StudentMapper.**class**);  *//一、使用selectByExample进行查询所有  //StudentExample studentExample=new StudentExample(); //相当于就是查询语句  //List<Student> list=studentMapper.selectByExample(studentExample);   //二、使用selectByExample进行条件查询* StudentExample studentExample=**new** StudentExample(); *//相当于就是查询语句  //添加查询条件* StudentExample.Criteria criteria=studentExample.createCriteria();  criteria.andAgeLessThan(**new** Byte(**"30"**)); *//拼sql* criteria.andNameLike(**"%王%"**);   List<Student> list=studentMapper.selectByExample(studentExample);  *//关闭sqlSession* MyBatisUtil.*closeSession*();   *//显示结果* System.***out***.println(**"学号\t姓名\t年龄\t性别\t生日\t年级"**);  list.forEach(s -> {  System.***out***.println(s.getXh()+**"\t"**+s.getName()+**"\t"**+s.getAge()+**"\t"**+s.getSex()+**"\t"**+s.getBirthday()+**"\t"**+s.getGid());  });  } |