# Lab 3 Annotation and Reflection.

Due date: April 10. 2020

## 实验目的

使用注解和反射来完成动态Sql编程

## 实验要求

1. **提供用户表：user**

表中包含字段：

id，用户名，性别，邮箱，电话等信息。

1. **要求通过注解和反射的方式封装一个小型的sql操作类，可以通过对应的方法生成增、删、改、查等操作的SQL语句。**
2. **要求实现注解**：

@Column：用来标注每个field对应的表中的字段是什么  
@Table：用来标记表的名字

1. **例子：**

（1）创建User类

|  |
| --- |
| package cn.tju.edu.advjava.reflection.entity; import cn.tju.edu.advjava.reflection.annotation.\*; /\*\* @author liebes \*/ @Table("user") public class User{  @Column("id")  private int id;  @Column("username")  private String username;  @Column("age")  private int age;   @Column("email")  private String email;   @Column("telephone")  private String telephone; } |

（2）创建接口

|  |
| --- |
| package cn.tju.edu.advjava.reflection.sql; import cn.tju.edu.advjava.reflection.entity.User; import java.util.List; public interface SqlUtil {   /\*\*  \* 根据传入的参数返回查询语句  \* @param user  \* @return 返回查询语句  \*/  String query(User user);   /\*\*  \* 根据传入的参数返回插入语句  \* @param user  \* @return 返回插入语句  \*/  String insert(User user);   /\*\*  \* 根据传入的参数返回插入语句  \* @param users  \* @return 返回插入语句  \*/  String insert(List<User> users);   /\*\*  \* 根据传入的参数返回删除语句（删除id为user.id的记录）  \* @param user  \* @return 返回删除语句  \*/  String delete(User user);  /\*\*  \* 根据传入的参数返回更新语句（将id为user.id的记录的其它字段更新成user中的对应值）  \* @param user  \* @return 返回更新语句  \*/  String update(User user); } |

实现对应的接口并进行测试：**注意这里只是需要大家把sql语句打印出来即可，不需要进行真正的数据库交互**

public class Main{  
 public static void main(String[] args) {  
 // initialize util  
 SqlUtil util = new SqlUtil();  
  
 // test query1  
 User user = new User();  
 user.setId(175);  
 System.out.println(util.query(user));  
 // print: SELECT \* FROM user WHERE id = 175  
  
 // test query2  
 user = new User();  
 user.setUsername("史荣贞");  
 System.out.println(util.query(user));  
 // print: SELECT \* FROM `user` WHERE `username` LIKE '史荣贞';  
  
 // test insert  
 user = new User();  
 user.setUsername("user");  
 user.setTelephone("12345678123");  
 user.setEmail("user@123.com");  
 user.setAge(20);  
 System.out.println(util.insert(user));  
 // print: INSERT INTO `user` (`username`, `telephone`, `email`, `age`) VALUES ('user', '12345678123', 'user@123.com', 20)  
   
 // test insert list  
 User user2 = new User();  
 user2.setUsername("user2");  
 user2.setTelephone("12345678121");  
 user2.setEmail("user2@123.com");  
 user2.setAge(20);  
 List<User> list = new ArrayList<>();  
 list.add(user);  
 list.add(user2);  
 System.out.println(util.insert(list));  
 // print: INSERT INTO `user` (`username`, `telephone`, `email`, `age`) VALUES ('user', '12345678123', 'user@123.com', 20), ('user2', '12345678121', 'user2@123.com', 20)  
   
 // test update  
 user = new User();  
 user.setId(1);  
 user.setEmail("change@123.com");  
 System.out.println(util.update(item));  
 // print: UPDATE `user` SET `email` = 'change@123.com' WHERE `id` = 1;  
   
 // test delete  
 user = new User();  
 user.setId(1);  
 System.out.println(util.delete(user));  
 // print: DELETE FROM `user` WHERE `id` = 1;  
 }  
}

## 实验报告

根据上述要求完成实验报告。实验报告中要体现注解和反射的实现细节，以及最后的测试输出。

## 注意事项

本实验只要求输出sql语句即可，如有余力可以使用jdbc进行真正的数据库交互操作。