电子科技大学信息与软件工程学院

**实 验 报 告**

学 号 201822090530

姓 名 李 欣 芮

（实验） 课程名称**Android系统结构与应用编程**

理论教师 饶 云 波

实验教师 饶 云 波

**电子科技大学教务处制表**

**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名：李欣芮 学号：201822090530 指导教师：饶云波**

**实验地点：电子科技大学主楼314 实验时间：2019.03.10**

**一、实验名称：Java核心技术及数据库操作**

**二、实验学时：4学时**

**三、实验目的：**

* 熟悉Java核心技术:类,接口,异常处理；
* 完成网络通信中Java的TCP/IP协议；
* 熟悉JDBC的数据库连接。

**四、实验原理：**

MySql的安装和配置

Java核心技术:类,接口,异常处理

**五、实验内容：**

1.完成MySql的安装和配置

2.完成数据库的连接和基本操作

3.完成Java核心技术:类,接口,异常处理

(1)编写Java程序,创建类Number，通过类中的方法count可得到任意两个数相乘的结果，并在调用该方法的主方法中使用try-catch语句捕捉可能发生的异常。

(2)编写Java程序，获得指定端口的主机名、主机地址和主机地址。

(3)创建类SearchEmp，实现查找数据表tb\_emp中销售部的所有成员的功能。

(4)编写程序，实现向数据表tb\_stu中添加数据的功能，要求姓名为“李某”，性别是“女”，出生日期是 “1999-10-20”。

(5)编写程序，实现删除出生日期在“2010-01-01”之间的学生的功能。

**六、实验器材（设备、元器件）：**

PC， 软件JDK包和Eclipse，MySql等。

**七、实验步骤：**

**1.下载并安装MySql5.7,启动数据库的MySql-server数据库服务，创建数据库。**

**2.打开Eclipse，创建新项目，在biud-path中引入下载好的mysql-connector-java-8.0.13.jar包，将其导入。**

**3.编写程序，连接数据库，编写写数据库语言，进行对数据库的读写访问。**

**4.打印实验结果并分析。**

**八、实验结果与分析（含重要数据结果分析或核心代码流程分析）**

1.编写Java程序,创建类Number，通过类中的方法count可得到任意两个数相乘的结果，并在调用该方法的主方法中使用try-catch语句捕捉可能发生的异常。

核心代码：

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Number n = **new** Number();

**while**(**true**) {

**try** {

n.count();

}**catch**(Exception e) {

System.***out***.println("异常："+e.getMessage());

**break**;

}

}

}

首先初始化Number对象，调用成员方法count,为了捕获异常在外面加了try-catch函数，当产生异常时会打印异常的消息。

结果分析：

请输入第一个数：5555

请输入第二个数：66.222

-----------------------Hello Java------------------------

66.222 x 5555.0 = 367863.22

请输入第一个数：444

请输入第二个数：dd555

异常：null

2. 编写Java程序，获得指定端口的主机名、主机地址和主机地址。

**try** {

// try语句块捕捉可能出现的异常

ip = InetAddress.*getLocalHost*(); // 实例化对象

String localname = ip.getHostName(); // 获取本机名

String localip = ip.getHostAddress(); // 获取本IP地址

System.***out***.println("本机名：" + localname);// 将本机名输出

System.***out***.println("本机IP地址：" + localip); // 将本机IP输出

} **catch** (UnknownHostException e) {

e.printStackTrace(); // 输出异常信息

}

3. 创建类SearchEmp，实现查找数据表tb\_emp中销售部的所有成员的功能。

**public** Object query(T t) {

ArrayList<T> res = **new** ArrayList<>();

HashMap<String, Object> obj = t2obj(t);

Class<?> c = t.getClass();

//字段名

ArrayList<String> nameList = **new** ArrayList<>();

Field[] fields = c.getFields();

**for**(Field field : fields) {

nameList.add(field.getName());

}

String names = StringUtils.*join*(nameList, ",");

//表名字

String tableName = c.getSimpleName().toLowerCase();

//条件

String condition = "";

//限制

String limit = "";

**for** (HashMap.Entry<String, Object> entry : obj.entrySet()) {

**if**(entry.getValue()!=**null**) {

**if**(entry.getKey() == "limit") {

limit = entry.getValue().toString();

}**else** {

**if**(entry.getValue() **instanceof** String) {

condition = String.*format*("%s='%s'", entry.getKey(), entry.getValue());

}**else** **if**(entry.getValue() **instanceof** Date){

SimpleDateFormat sdf = **new** SimpleDateFormat("yyyy-MM-dd");

condition = String.*format*("%s='%s'", entry.getKey(), sdf.format(entry.getValue()));

}**else** {

condition = String.*format*("%s=%s", entry.getKey(), entry.getValue());

}

}

}

}

**if**(names == "")names = "\*";

**if**(condition!="")condition = " WHERE "+condition;

**if**(limit!="")limit = " LIMIT "+limit;

String sql =String.*format*(query, names, tableName, condition, limit);

System.***out***.println(sql);

**try** {

Class.*forName*("com.mysql.cj.jdbc.Driver");

conn=(Connection)DriverManager.*getConnection*(url,username,password);

ps = conn.prepareStatement(sql);

rs = ps.executeQuery(sql);

res= **this**.rs2obj(rs, c);

} **catch** (ClassNotFoundException | SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}**finally** {

**this**.close();

}

**if**(res.size()==1)**return** res.get(0);

**return** res;

}

输入相应的要查询的对象，并进行一定的约束，通过反射可以获取对象的字段，要查询的数据库查询语句，返回结果集，并打印结果

结果分析：

SELECT \* FROM emp

id : 1

name : 张三

sex : 男

id : 2

name : 李四

sex : 男

id : 3

name : 麻子

sex : 女

4. 编写程序，实现向数据表tb\_stu中添加数据的功能，要求姓名为“李某”，性别是“女”，出生日期是 “1999-10-20”。

**public** **void** insert(T t) {

String tableName = t.getClass().getSimpleName().toLowerCase();

HashMap<String, Object> obj = t2obj(t);

System.***out***.println(obj);

ArrayList<String> nameList = **new** ArrayList<>();

ArrayList<Object> valueList = **new** ArrayList<>();

**for** (HashMap.Entry<String, Object> entry : obj.entrySet()) {

**if**(!entry.getKey().equals("id")) {

nameList.add(entry.getKey());

**if**(entry.getValue() **instanceof** String) {

valueList.add(String.*format*("'%s'", entry.getValue()));

}**else** **if**(entry.getValue() **instanceof** Date) {

SimpleDateFormat sdf = **new** SimpleDateFormat("yyyy-MM-dd");

valueList.add(String.*format*("'%s'", sdf.format(entry.getValue())));

}**else** {

valueList.add(entry.getValue());

}

}

}

String names = StringUtils.*join*(nameList, ",");

String values = StringUtils.*join*(valueList, ",");

System.***out***.println(names);

System.***out***.println(values);

String insertSql =String.*format*(insert, tableName, names, values);

System.***out***.println(insertSql);

**this**.update(insertSql);

}

本段代码，首先利用反射加载类名和字段，对id主键进行判定分别对字段的类型进行审查，然和将对应的字段值补充，将记录插入到数据库中。

代码结果：

{born=Wed Oct 20 00:00:00 CST 1999, sex=女, name=李某, id=null}

born,sex,name

'1999-10-20','女','李某'

INSERT INTO tb\_stu (born,sex,name) VALUES ('1999-10-20','女','李某')



5. 编写程序，实现删除出生日期在“2010-01-01”之间的学生的功能。

**public** **void** delete(T t) {

String tableName = t.getClass().getSimpleName().toLowerCase();

HashMap<String, Object> obj = **this**.t2obj(t);

ArrayList<String> constraintList = **new** ArrayList<>();

**for**(HashMap.Entry<String, Object> entry : obj.entrySet()) {

**if**(entry.getValue()!=**null**) {

**if**(entry.getValue() **instanceof** String) {

constraintList.add(String.*format*("%s='%s'", entry.getKey(), entry.getValue()));

}**else** **if**(entry.getValue() **instanceof** Date){

SimpleDateFormat sdf = **new** SimpleDateFormat("yyyy-MM-dd");

constraintList.add(String.*format*("%s='%s'", entry.getKey(), sdf.format(entry.getValue())));

}**else** {

constraintList.add(String.*format*("%s=%s", entry.getKey(), entry.getValue()));

}

**break**;

}

}

String constaints = StringUtils.*join*(constraintList, ",");

String deleteSql = String.*format*(delete, tableName, constaints);

System.***out***.println(constraintList);

System.***out***.println(constaints);

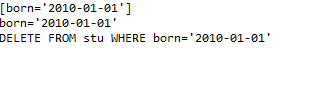
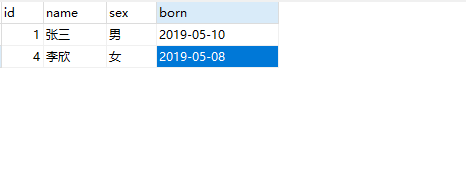
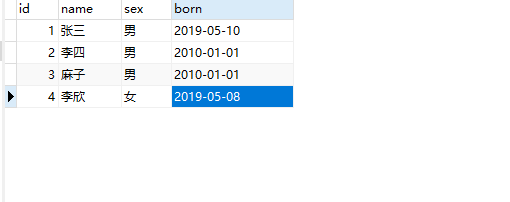
System.***out***.println(deleteSql);

**this**.update(deleteSql);

}

本段代码实现了对泛型的反射代理，获取对象并执行删除语句。步骤加载数据库连接驱动，将数据库语言加载到prestatement上，执行语句。

代码结果：



**九、总结及心得体会：**

**本次实验掌握了java的数据库的编程技术，了解到整个数据库编程的技术细节，并在此技术上加上了泛型和反射来增加其数据库的通用性。**

**十、对本实验过程及方法、手段的改进建议：**

**该实验的改进可以增加关于设计模式的概念进去。**

**报告评分：**

**指导教师签字：**