
数据库算法实验报告

一、实验目的

1. 理解数据库相关概念
2. 理解 JDBC API 的工作原理
3. 掌握 java.sql 包中常用的类和接口
4. 掌握编写 Java 数据库程序的关键步骤

二、相关知识

通过 JDBC 实现数据库编程的步骤：

1. 加载相应数据库的驱动
2. 连接数据库
3. 得到相应的 SQL 语句对象
4. 向数据库提交 SQL 语句对象，从数据库取回记录集对象
5. 在程序中处理记录集
6. 关闭连接

三、实验内容

1. 完成 Java 与 MYSQL 数据库的连接并测试
2. 构建一个具有图形界面的数据管理系统

四、实现思路

4.1 完成 Java 与 MYSQL 数据库的连接并测试

GUIMySQL.java

1. 首先创建一个学生类实现三个属性：id, name, math

```

class Student implements Serializable{
    /**
     *
     */
    private static final long serialVersionUID = 1L;
    private int id;
    private String name;
    private int math;
    public Student() {

    }
    public Student(int id, String name, int math) {
        this.id = id;
        this.name = name;
        this.math = math;
    }
    public int getId() {
        return this.id;
    }
    public String getName() {
        return this.name;
    }
    public int getMath() {
        return this.math;
    }
    public void setId(int id) {
        this.id = id;
    }
    public void setName(String name) {
        this.name = name;
    }
    public void setMath(int math) {
        this.math = math;
    }
}

```

图 1 student 类

2. 初始化调用的数据库的名字和表的名字并初始化 sql 连接类

```

class Sql {
    static Connection connection;
    static String databases = "data";
    static String table = "student1";
}

```

图 2 静态变量

3. 连接 mysql 数据库的静态函数 getConnection()

通过 DriverManager.getConnection(url,user, password)连接到对应的数据库中，并返回连接对象 con。

```
/*
 * 方法连接到实例数据库
 */
public static Connection getConnection(Connection con) throws SQLException, java.lang.ClassNotFoundException{
    String url = "jdbc:mysql: // localhost:3306/" + databases; //连接mysql中的test数据库
    Class.forName("com.mysql.jdbc.Driver");
    String userNameString = "root";
    String password = "123456";
    con = DriverManager.getConnection(url, userNameString, password);
    return con;
}
```

图 3 连接数据库的函数

4. 实现对表中数据的增删改查

- 增加数据，将学生对应的三个属性传入，插入成功则打印插入数据成功
- 删除数据，通过主键 id 来进行删除，删除成功则打印删除成功
- 更新数据，考虑到同名同姓的情况，所以根据 id 和姓名一起进行更新
- 查询数据，通过 id 进行查询

```
// 增加数据
public static void add(int id,String name,int math, Statement sql){
    String query = "insert "+table+" values("+id+", '"+name+"', "+math+")";
    System.out.println(query);
    try{
        sql.execute(query);
        System.out.println("插入数据成功!");
    }catch(Exception e){
        e.printStackTrace();
    }
}

// 删除数据
public static void delete(int id,Statement sql){
    String query = "delete from "+table+" where(id="+id+")";
    System.out.println(query);
    try{
        sql.execute(query);
        System.out.println("删除数据成功!");
    }catch(Exception e){
        e.printStackTrace();
    }
}
```

图 4 增删改查的部分函数

5. Main 函数中实现对三个学生的例化，数据库的连接，表的创建，数据的增删改查

```
Student a = new Student(1, "AAA", 99);
Student b = new Student(2, "BBB", 77);
Student c = new Student(3, "CCC", 65);
Connection con = getConnection(connection);
Statement sql = con.createStatement();
sql.execute("drop table if exists student1");
sql.execute("create table student1("+
    "id int not null auto_increment,"+
    "name varchar(20) not null default 'name',"
    "math int not null default 60,"+
    "primary key(id))");
add(a.getId(), a.getName(), a.getMath(), sql);//添加数据
add(b.getId(), b.getName(), b.getMath(), sql);
add(c.getId(), c.getName(), c.getMath(), sql);
add(4, "DDD", 100, sql);
delete(1, sql);//删除数据
update(4, "DDD", 60, sql);//更新数据
research(3, sql);//单独查询数据
String query = "select * from student1";
ResultSet result = sql.executeQuery(query);
System.out.println("Student1 表数据如下: ");
System.out.println("-----");
System.out.println("学号 姓名 数学成绩");
System.out.println("-----");
int number;
String name;
int math;
while(result.next()) {
    number = result.getInt("id");
    name = result.getString("name");
    math = result.getInt("math");
    System.out.println(number+"    "+name+"    "+math);
}
sql.close();
con.close();
}catch (java.lang.ClassNotFoundException e) {
    System.err.println("ClassNotFoundException:"+e.getMessage());
}catch (SQLException ex) {
    System.err.println("SQLException:"+ ex.getMessage());
}
```

图 6 测试模块

6. 输出结果:

```

insert student1 values(1,'AAA',99)
插入数据成功!
insert student1 values(2,'BBB',77)
插入数据成功!
insert student1 values(3,'CCC',65)
插入数据成功!
insert student1 values(4,'DDD',100)
插入数据成功!
delete from student1 where(id=1)
删除数据成功!
update student1 set math=60 where id=4 and name = 'DDD'
更新数据成功!
Select * from student1 where id=3
查询的结果:
3   CCC   65
查询数据成功!

```

图 7 增删改查的结果

Student1 表数据如下:

学号 姓名 数学成绩

2	BBB	77
3	CCC	65
4	DDD	60

图 8 打印 student1 中存储的数据

我们可以看到插入 d 的 DDD 的 math 成绩从 100 更新到了 60
Id 为 1 的数据被删除了

7. 在 MYSQL 的控制台查看真实数据库中表的数据与输出是否一致

```

mysql> describe student1;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| id    | int           | NO   | PRI | NULL    | auto_increment |
| name  | varchar(20)   | NO   |     |         |                |
| math  | int           | NO   |     |         |                |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> select * from student1;
+----+-----+-----+
| id | name | math |
+----+-----+-----+
| 2  | BBB  | 77   |
| 3  | CCC  | 65   |
| 4  | DDD  | 60   |
+----+-----+-----+
3 rows in set (0.00 sec)

```

可以看到结果一致，实验成功。

4.2 具有图形界面的数据库管理系统

1. 参考 https://blog.csdn.net/weixin_44251578/article/details/85280189 进行的图形界面数据库的实现

五、实验收获

六、完整代码