电子科技大学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

软件技术基础综合课程设计

设计报告

学生姓名：李逢君

学 号：**2016060601010**

指导教师：林劼，张栗粽

实验地点：主楼 **b1-302**

实验时间：**2018.7.11**

目录

一、实验室名称： 主楼 b1-302

二、实验项目名称： 互联网+智慧物流质询系统

三、实验学时：32

四、实验原理：

实验一：

通过物品信息计算物流方式以及发货顺序，然后用树这种数据结构存储这些物品信息，用邻接矩阵存储路线信息，并利用 dijstra 算法求最短路。

实验二：

利用文件锁使得进程间互斥对文件进行访问，实现同时写与同时读写的互斥操作。保证文件的一致性。

实验三：

通过词法分析器，语法分析器以及语义分析器对输入的 sql 进行解析并转化成相应的中间代码。

实验四：

整合前三个实验，使之成为一个系统，并能通过它对数据进行增删查改以及物流物品信息查看。

五、实验目的：

实验一：

(1) 根据物品信息综合计算物流物品的优先级别，根据物流优先级别排序物流物品，根据排序结果对物流物品进行逐个发货。

(3) 根据物流物品的物流条件信息，归类物流物品到物流方案类型，物流方案类型可包括：价格最小物流方案，时间最短物流方案、综合最优方案、航空物流方案等。并运用树型结构存储所有的物

流物品到划分的物流方案中。

(4) 根据给定的物流节点信息，计算各类物流方案下的物流最短路径

(5) 根据物流最短路径，物流方案和物流优先级发送货物

实验二：

(1)编程实现文件中记录数据的存储、读写和记录的简单查询与索引查询函数。能够实现单用户和进行对文件数据的写入与查询。

(2)设计实现数据表的文件存储方式，能在文件中存储多张数据表，并写入和查询指定　数据表中的记录。

(3 实现多进程对文件记录的互斥写入与查询访问,保证记录数据的一致性。

(4)基于锁机制保障多用户对文件中记录数据的写如与查询一致性操作。

实验三：

(1)构建词法、语法及语义分析程序实现部分 SQL 语句的解析，包括 Select 语句， Insert 语句和 Update 语句，创建表语句的解析。

(2)构建相应的语义子程序。

(3)将于语义子程序对接底层实验 2 中所实现的各个数据操作函数,实现增删查改功能。

实验四：

(1)结合实验 3,2 构建物流节点信息表,实现物流节点信息的数据库存储

(3)结合实验 3,2，构建物品信息表，实现物品信息的存储

(4)结合实验 3 以 SQL 语句，对物流节点信息进行增删改查

(5)结合实验 3 以 SQL 语句，对物品信息进行增删改查

(6)结合实验 1，实现物品的优先级排序和物流方案分类

(7)节点信息会动态变化，因此结合实验 1，每个物品需要动态计算物流最短路径的实现。

（8）模拟物品的物流状态，用户可以对物件的物流状态进行查询。

六、实验内容：

实验一：最优物流路线计算实验

数据存储，对于物流物品及不同方案路径的信息使用 excel 进行存储，使用 Java 的 poi 包进行 i/o 操作。

数据生成，对于物流物品的信息设置包括 id、名称、发货地、目的地、类型、客户等级、收货时间以及一个标志位作为软删除的标志。id 的属性从 1 开始递增，物品名称使用随机数 a-z 进行生成，发货地和目的地使用随机数 A-G 进行生成，类型和客户等级使用一定范围的随机数字生成，收货时间也是使用相应公式随机生成 2018-4-1 至 2018-4-30 的范围，标志位 1 则认为物品存在，为 0 则认为物品删除，生成物流物品信息共 1000 条；对于不同方案的路径信息也使用随机数 0-9 进行生成，存储为多个二维邻接矩阵，对于路径为 0 的节点使用 10000 来代替，代表不可达，根据价格最小物流方案，时间最短物流方案、综合最优方案、航空物流方案四个方案共生成 4 个邻接矩阵。

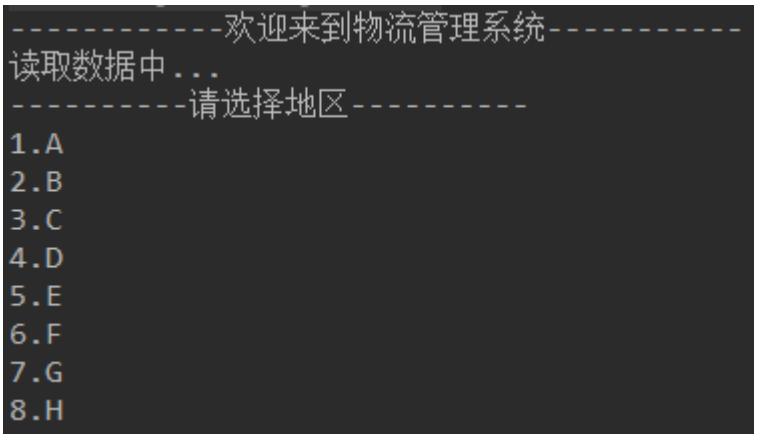
数据的预处理，考虑到发货地和目的地随机生成可能导致两者相同，将相同的随机一方改写成 H，更新路径矩阵。

id	name	origin	destination	type	client_level	date	flag
10	y	G	E	2	3	2018-4-24	1
11	h	C	F	2	6	2018-4-30	1
12	h	H	F	2	1	2018-4-23	1
13	e	C	A	4	4	2018-4-25	1
14	r	F	H	3	2	2018-4-23	1
15	l	F	G	4	6	2018-4-29	1
16	n	A	E	1	2	2018-4-29	1
17	b	F	G	2	3	2018-4-24	1
18	u	H	F	2	7	2018-4-26	1
19	q	G	C	3	3	2018-4-28	1
20	o	A	F	2	1	2018-4-25	1
21	f	E	A	2	3	2018-4-26	1
22	c	G	F	1	2	2018-4-27	1
23	z	A	D	2	7	2018-4-23	1
24	n	F	H	4	3	2018-4-29	1
25	z	E	F	2	1	2018-4-29	1
26	n	D	A	2	2	2018-4-29	1
27	v	C	F	3	5	2018-4-28	1
28	t	A	B	3	6	2018-4-30	1
29	c	B	E	4	2	2018-4-25	1
30	w	E	B	2	4	2018-4-27	1

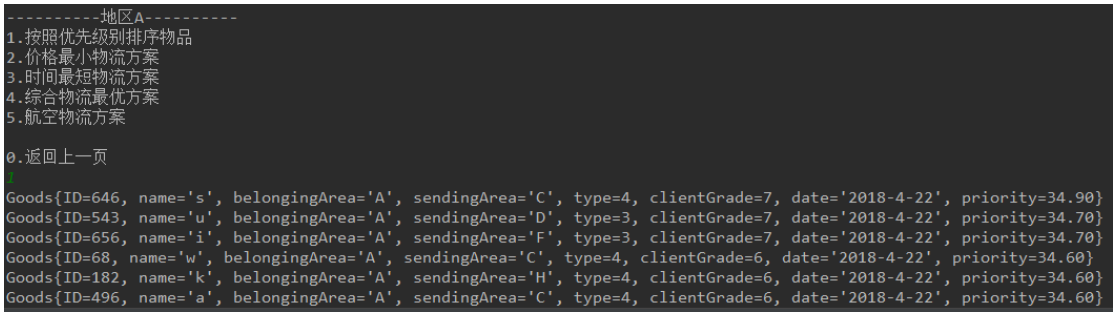
	A	B	C	D	E	F	G	H
A	0	7	8	10000	8	4	8	3
B	7	0	4	10000	10000	7	8	5
C	8	4	0	7	7	1	8	6
D	10000	10000	7	0	8	4	8	8
E	8	10000	7	8	0	10000	9	2
F	4	7	1	4	10000	0	9	10000
G	8	8	8	8	9	9	0	3
H	3	5	6	8	2	10000	3	0

具体功能实现：

物品的优先级计算，使用线性结构存储读取的数据。根据物品所属地对地区进行选择。



根据物流物品的类型、客户等级、收货时间赋予一定的权重计算物品的优先级，并且按照降序对物品的优先级进行排序。



根据物流物品的类型，将物品归类为不同的物流方案，将 type=1 的物品归类到价格最小物流方案，将 type=2 的物品归类到时间最短物流方案，以此类推。并且可以根据优先级输出该方案物品。

```
1.按优先级输出该方案的物品
2.输出最短路径
3.根据物品ID发货

0.返回上一页
1
Goods{ID=794, name='x', belongingArea='A', sendingArea='C', type=1, clientGrade=7, date='2018-4-22', priority=34.30}
Goods{ID=162, name='d', belongingArea='A', sendingArea='C', type=1, clientGrade=7, date='2018-4-23', priority=33.90}
Goods{ID=373, name='x', belongingArea='A', sendingArea='C', type=1, clientGrade=6, date='2018-4-23', priority=33.60}
Goods{ID=262, name='q', belongingArea='A', sendingArea='D', type=1, clientGrade=7, date='2018-4-24', priority=33.50}
Goods{ID=331, name='v', belongingArea='A', sendingArea='D', type=1, clientGrade=7, date='2018-4-24', priority=33.50}
Goods{ID=291, name='l', belongingArea='A', sendingArea='E', type=1, clientGrade=3, date='2018-4-22', priority=33.10}
Goods{ID=852, name='y', belongingArea='A', sendingArea='G', type=1, clientGrade=5, date='2018-4-24', priority=32.90}
Goods{ID=939, name='f', belongingArea='A', sendingArea='F', type=1, clientGrade=5, date='2018-4-24', priority=32.90}
```

通过 Dijkstra 算法计算各类物流方案下的物流最短路径。

```
1.按优先级输出该方案的物品
2.输出最短路径
3.根据物品ID发货

0.返回上一页
2
从A出发到A的最短路径为：A-->A，最短距离为：0
从A出发到B的最短路径为：A-->B，最短距离为：7
从A出发到C的最短路径为：A-->F-->C，最短距离为：5
从A出发到D的最短路径为：A-->F-->D，最短距离为：8
从A出发到E的最短路径为：A-->H-->E，最短距离为：5
从A出发到F的最短路径为：A-->F，最短距离为：4
从A出发到G的最短路径为：A-->H-->G，最短距离为：6
从A出发到H的最短路径为：A-->H，最短距离为：3
=====================================
```

可根据物品的 ID 进行发货。

```
1.按优先级输出该方案的物品
2.输出最短路径
3.根据物品ID发货

0.返回上一页
3
请输入物品ID发货，输入0返回上一层：
16
Goods{ID=16, name='n', belongingArea='A', sendingArea='E', type=1, clientGrade=2, date='2018-4-29', priority=30.00}
从A出发到E的最短路径为：A-->H-->E，最短距离为：5
```

实验二：多进程多用户文件一致性读写访问设计实现

实验二在上述实验的表格上实现增删改查及多用户文件一致性读写访问。主要使用 Java 的 poi 实现对表格的增删查改，以及使用 java.nio.channels.FileChannel 类实现进程间通信以及 java.nio.channels.FileLock 类实现文件加锁，多用户访问的实现主要在逻辑上，即每次更改数据前都要读取一次最新的数据到内存中，而不能在程序开始时将数据读取到内存中，以后保持不变，这是不可取的。

文件读取

```
----------------文件读取中----------------

请选择表格：
1. goodsList
2. route1
3. route2

0. 退出系统
```

查询数据

```
===============  查询数据 ===============
1. 查询所有数据
2. 查询所有列名
3. 查询指定列
4. 根据索引查询

0. 返回上一层
1
id name origin destination type client_level date
2 j D F 3 4 2018-4-27
4 2 3 4 5 1 2
5 b F D 4 4 2018-4-26
6 n G H 4 7 2018-4-24
7 j F E 3 5 2018-4-26
8 r E G 1 6 2018-4-29
9 f G D 4 5 2018-4-29
10 y G E 2 3 2018-4-24
```

增加数据

```
===============  增加数据 ===============
name：
1
origin：
2
destination：
3
type：
4
client_level：
5
date：
6

添加成功!
```

```
1030 5 5 5 5 5 5
1031 1 1 1 1 1 1
1032 1 1 1 1 1 1
1033 2 3 2 3 2 3
1034 3 3 3 3 3 3
1035 1 1 1 1 1 1
1037 1 2 3 4 5 6

===============  查询数据 ===============
1. 查询所有数据
2. 查询所有列名
3. 查询指定列
4. 根据索引查询

0. 返回上一层
```

修改数据

```
===============  修改数据 ===============
请输入需要修改的数据索引，输入0返回上一层：

1037

id name origin destination type client_level date
1037 1 2 3 4 5 6
name：
6
origin：
5
destination：
4
type：
3
client_level：
2
date：
1
修改成功!
```

```
1030 5 5 5 5 5 5
1031 1 1 1 1 1 1
1032 1 1 1 1 1 1
1033 2 3 2 3 2 3
1034 3 3 3 3 3 3
1035 1 1 1 1 1 1
1037 6 5 4 3 2 1

===============  查询数据 ===============
1. 查询所有数据
2. 查询所有列名
3. 查询指定列
4. 根据索引查询

0. 返回上一层
```

删除数据，数据库中标志位置 0

```
===============  删除数据 ===============
请输入需要删除的数据索引，输入0返回上一层：

1037

id name origin destination type client_level date
1037 6 5 4 3 2 1
删除成功!
```

```
1030 5 5 5 5 5 5
1031 1 1 1 1 1 1
1032 1 1 1 1 1 1
1033 2 3 2 3 2 3
1034 3 3 3 3 3 3
1035 1 1 1 1 1 1

===============  查询数据 ===============
1. 查询所有数据
2. 查询所有列名
3. 查询指定列
4. 根据索引查询
```

id	name	origin	destinatior	type	client_level	date	flag
1034	3	3	3	3	3	3	1
1035	1	1	1	1	1	1	1
1036	2	2	2	2	2	2	0
1037	6	5	4	3	2	1	0

多进程多用户文件一致性读写的测试使用主动的程序停滞 10s 来模拟高并发的写文件。

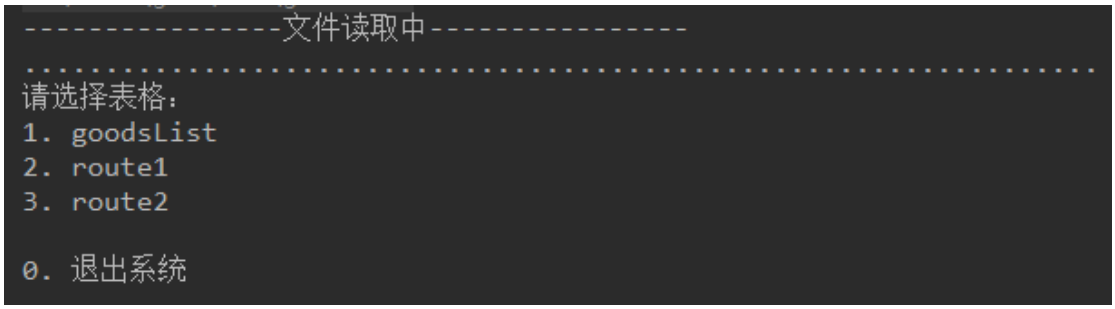
```java
private static void saveFile() {
    try {
        file.createNewFile();
        outStream = FileUtils.openOutputStream(file);   // 将excel存盘
        FileChannel channel = outStream.getChannel();
        try {
            //方法一
            lock = channel.lock();
            workbook.write(outStream);
            Thread.sleep( millis: 10000); //for test

            //方法二
            //lock = channel.tryLock();
            //if(lock != null){
            //  do something..
            //}
        } catch (IOException e) {
            e.printStackTrace();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        if (null != lock) {
            try {
                lock.release();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
        if (outStream != null) {
            try {
                outStream.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
}
```

此时启动第一个程序进行写文件，第二个程序进行文件操作的时候会报以下错误。

```
java.io.FileNotFoundException: src\main\resources\data.xls (另一个程序正在使用此文件，进程无法访问。)
    at java.io.FileOutputStream.open0(Native Method)
    at java.io.FileOutputStream.open(FileOutputStream.java:270)
    at java.io.FileOutputStream.<init>(FileOutputStream.java:213)
    at org.apache.commons.io.FileUtils.openOutputStream(FileUtils.java:367)
    at org.apache.commons.io.FileUtils.openOutputStream(FileUtils.java:326)
    at Main.saveFile(Main.java:476)
    at Main.dataAdd(Main.java:145)
    at Main.sheetSelector(Main.java:93)
    at Main.main(Main.java:45)
java.nio.channels.ClosedChannelException
    at sun.nio.ch.FileLockImpl.release(FileLockImpl.java:58)
    at Main.saveFile(Main.java:496)
    at Main.dataAdd(Main.java:145)
    at Main.sheetSelector(Main.java:93)
    at Main.main(Main.java:45)
```

对该错误进行捕捉，打印“…”，并主动停等 100ms 再进行文件操作，直到文件操作成功为止。



实验三：SQL 解析器设计实现

SQL 解析器的设计是基于 JSqlParser 进行实现，并与实验 2 的接口进行对接。

Select 语句

select id, name, origin from goodsList where id > 5 and name <= b and origin >= D

```
select id, name, origin from goodsList where id > 5 and name <= b and origin >= D
id name origin
51 a F
64 a E
80 a F
96 b H
503 a D
513 a D
571 b F
598 b D
607 a D
```

select * from goodsList where id > 5 and name <= b and origin >= D

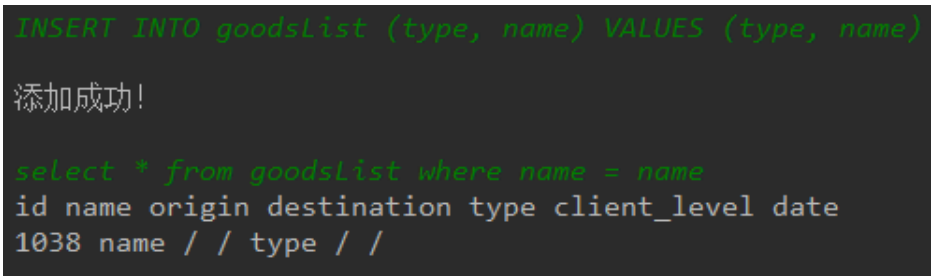
```
select * from goodsList where id > 5 and name <= b and origin >= D
id name origin destination type client_level date
51 a F C 2 1 2018-4-28
64 a E A 4 1 2018-4-23
80 a F C 1 1 2018-4-29
96 b H A 4 1 2018-4-27
503 a D F 2 6 2018-4-29
513 a D H 1 4 2018-4-28
571 b F C 1 7 2018-4-29
598 b D G 1 7 2018-4-30
607 a D A 1 6 2018-4-22
619 a G A 1 3 2018-4-23
```

Insert 语句

INSERT INTO goodsList VALUES ('Gates', 'Bill', 'Xuanwumen 10', 'Beijing', '33', '33')

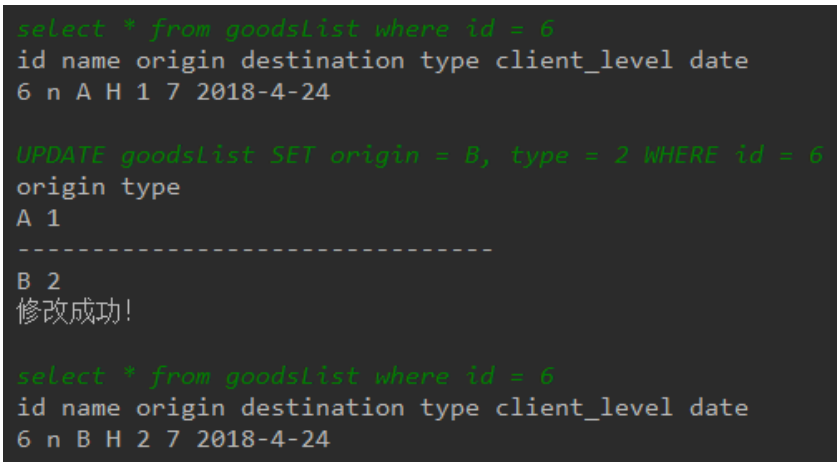
```
INSERT INTO goodsList VALUES ('Gates', 'Bill', 'Xuanwumen 10', 'Beijing', '33', '33')
添加成功!
select * from goodslist where name = 'Gates'
id name origin destination type client_level date
1037 'Gates' 'Bill' 'Xuanwumen 10' 'Beijing' '33' '33'
```

INSERT INTO goodsList (type, name) VALUES (type, name)



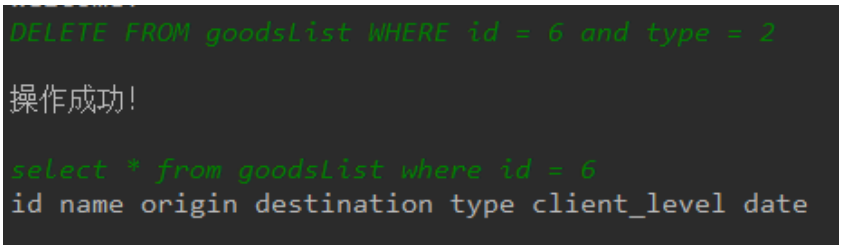
Update 语句

UPDATE goodsList SET origin = B, type = 2 WHERE id = 6



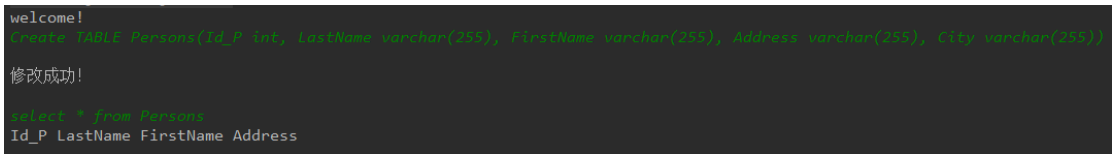
Delete 语句

DELETE FROM goodsList WHERE id = 6 and type = 2



创建表 createTable 语句

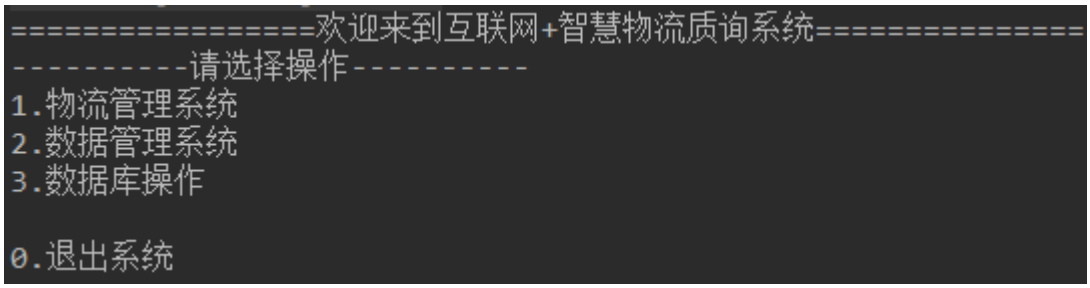
Create TABLE Persons(Id_P int, LastName varchar(255), FirstName varchar(255), Address varchar(255), City varchar(255))



实验四：互联网+智慧物流质询系统设计实现

实验四将之前的三个实验进行合并，由于之前三个实验都是使用松耦合的方

式进行开发，并且前三个实验实现的功能与第四个完全符合，因此实验四将前三个实验分为物流管理系统、数据管理系统以及数据库操作三个模块，并通过调用不同模块的 main 函数进行操作。



因逻辑基本相同，故不再赘述。

七、实验器材（设备、元器件）：

JAVA 及相关类包，PC 电脑，Windows 操作系统，IDEA 集成开发环境，

八、数据结构与程序：

源代码地址: https://github.com/543877815/ruanji