

基于 JAVA 和 MySQL 的邮件检索

JAVA 作业 2-1613415-潘巧巧

目录

一、	整体实现基本情况.....	2
二、	SaveDataToMysql.java.....	2
1、	基本情况.....	2
2、	getFiles()	3
3、	getData().....	3
4、	InsertMysql().....	5
三、	SearchData.java.....	6
1、	程序效果演示.....	6
2、	基本情况.....	8
3、	SearchMysqlInfo()	9
4、	ShowResult()	10
四、	遇到的问题.....	12

一、整体实现基本情况

主要包含 2 个 java 文件。

- (1) SaveDataToMysql.java: 读取所有文件并进行关键信息的切割, 连接 mysql 数据库, 将路径 (Path)、标题 (Subject)、发件人 (From)、邮件内容 (Content) 存入 mysql 中。将 517401 封邮件的信息全部存入 mysql 中。

```
mysql> select count(*) from Email;
+-----+
| count(*) |
+-----+
| 517401   |
+-----+
1 row in set (10.59 sec)
```

MySQL 建表:

```
CREATE TABLE `Email` (
  `Path`      VARCHAR(500) NOT NULL,
  `Subject`   VARCHAR(500),
  `From`      VARCHAR(500),
  `Content`   LONGTEXT,
  PRIMARY KEY(`Path`));
```

- (2) SearchData.java: 用户输入检索类型和检索内容, 连接 mysql 后进行检索, 并将结果返回给用户。每次返回 10 条, 通过翻页展示。

二、SaveDataToMysql.java

1、基本情况

包含 3 个步骤

- (1) getFiles(): 获取所有邮件的绝对路径
- (2) getData(): 遍历所有邮件, 截取关键信息 (标题、发件人、邮件内容)。

(3) InsertMysql(): 数据插入 mysql 中。

```
public static void main(String[] args) {  
    List<File> files = getFiles( path: "data\\maildir");  
    List<DATA> dates = getData(files);  
    InsertMysql(dates);  
}
```

```
// 获取path下所有文件夹  
private static List<File> getFiles(String path) {...}  
// 切割各个文件获取关键信息  
private static List<DATA> getData(List<File> files) {...}  
// 数据插入mysql中  
private static void InsertMysql (List<DATA> dates){...}
```

2、 getFiles()

将 path 下所有文件的绝对路径保存在一个 List 中并返回

```
// 获取path下所有文件夹  
private static List<File> getFiles(String path) {  
    File root = new File(path);  
    List<File> files = new ArrayList<File>();  
    if (!root.isDirectory()) {  
        files.add(root);  
    } else {  
        File[] subFiles = root.listFiles();  
        assert subFiles != null;  
        for (File f : subFiles) {  
            files.addAll(getFiles(f.getAbsolutePath()));  
        }  
    }  
    return files;  
}
```

3、 getData()

该函数遍历所有邮件,截取有用信息,存成一个 DATA 类的示例,最后返回一个 List<DATA>

```

class DATA{
    String Path;
    String Subject;
    String Author;
    String Content;
    DATA()
    {
        this.Path = "";
        this.Subject = "";
        this.Author = "";
        this.Content = "";
    }
}

```

```

private static List<DATA> getData(List<File> files) {
    List<DATA> all_data = new ArrayList<DATA>();
    for (File cur_file : files) { // 遍历所有文件
        String cur_file_path = cur_file.getAbsolutePath();
        try {
            DATA temp_data = new DATA();
            temp_data.Path = cur_file_path;
            FileReader f = new FileReader(cur_file_path);
            BufferedReader buf = new BufferedReader(f);
            String s;
            boolean if_content = false; // 当前是否是正文部分
            StringBuilder temp_content = new StringBuilder(); // 正文部分记录
            while ((s = buf.readLine()) != null) {
                if (!if_content) { // 当前还没当正文部分
                    if (s.contains(":")) { // 如果当前行中包含了冒号
                        String[] tempString = s.split(":"); //按冒号切割
                        if (tempString.length >= 2) { // 切割后至少有 2 个部分
                            switch (tempString[0]) {
                                case "Subject":
                                    StringBuilder temp_subject = new StringBuilder();
                                    for (int i = 1; i < tempString.length; ++i) {
                                        temp_subject.append(tempString[i]);
                                    }
                                    temp_data.Subject = temp_subject.toString();
                                    break;
                                case "From":
                                    StringBuilder temp_from = new StringBuilder();
                                    for (int i = 1; i < tempString.length; ++i) {
                                        temp_from.append(tempString[i]);
                                    }

```

```

        temp_data.Author = temp_from.toString();
        break;
        case "X-FileName":
            if_content = true; //后面开始是正文部分
            break;
        default:
    }
    }
    }
    } else { // 正文部分
        temp_content.append(s);
    }
}
temp_data.Content = temp_content.toString();
f.close();
buf.close();
all_data.add(temp_data);
} catch (IOException e) {
    System.out.println("error: " + cur_file_path);
}
}
return all_data;
}

```

4、 InsertMysql()

将 List<DATA>中的数据插入 mysql 中。

```

private static void InsertMysql (List<DATA> dates){
    Connection conn = null;
    PreparedStatement stmt = null;
    try {
        // 注册 JDBC 驱动
        Class.forName(JDBC_DRIVER);
        // 打开链接
        conn = DriverManager.getConnection(DB_URL, USER, PASS);
        // 执行 sql 语句
        for (DATA data : dates) {
            String sql = "INSERT INTO `Email` VALUES (?, ?, ?, ?)";
            stmt = conn.prepareStatement(sql);
            stmt.setString(1, data.Path);
            stmt.setString(2, data.Subject);
            stmt.setString(3, data.Author);

```

```

        stmt.setString(4, data.Content);
        stmt.executeUpdate();
        System.out.println("InsertMysql: " + data.Path);
    }
    // 完成后关闭
    stmt.close();
    conn.close();
} catch (Exception se) {
    se.printStackTrace(); // 处理 JDBC 错误
}
finally {
    // 关闭资源
    try {
        if (stmt != null) stmt.close();
    } catch (SQLException ignored) {
    } // 什么都不做
    try {
        if (conn != null) conn.close();
    } catch (SQLException se) {
        se.printStackTrace();
    }
}
}
}

```

三、SearchData.java

1、 程序效果演示

(1) 用户输入检索类型，1 为按标题检索，2 为按发件人检索，3 为按邮件正文检索，4

为退出程序，其他输入提示输入错误，请求重新输入。

```

请输入检索类型
1 (标题) 2 (发件人) 3 (正文) 4 (退出)
5
输入错误!
请输入检索类型
1 (标题) 2 (发件人) 3 (正文) 4 (退出)
4
断开与目标 VM 的连接, 地址: '127.0.0.1:56389', transport: 'socket'

进程已结束, 退出代码 0

```

- (2) 用户输入检索内容，程序会展示搜索结果条数，并且 10 条为一页展示。当前处于第一页时，按 n 进入下一页；当前处于最后一页时，按 l 进入上一页；其他情况按 n 进入下一页，l 进入上一页。所有情况下 q 退出（返回第 1 步），不符合条件的错误输入报错。

```
请输入检索类型
1 (标题) 2 (发件人) 3 (正文) 4 (退出)
1
请输入检索内容:
Memorandum
共 274 条搜索结果，共 28 页，当前第 1 页
0: D:\Study\JAVA\project\java1\data\maildir\badeer-r\all_documents\102
1: D:\Study\JAVA\project\java1\data\maildir\badeer-r\california\4
2: D:\Study\JAVA\project\java1\data\maildir\badeer-r\discussion_threads\177
3: D:\Study\JAVA\project\java1\data\maildir\beck-s\all_documents\1812
4: D:\Study\JAVA\project\java1\data\maildir\beck-s\all_documents\1937
5: D:\Study\JAVA\project\java1\data\maildir\beck-s\all_documents\2174
6: D:\Study\JAVA\project\java1\data\maildir\beck-s\all_documents\2334
7: D:\Study\JAVA\project\java1\data\maildir\beck-s\curve_validation\3
8: D:\Study\JAVA\project\java1\data\maildir\beck-s\curve_validation\4
9: D:\Study\JAVA\project\java1\data\maildir\beck-s\curve_validation\5
输入 n 进入下一页，输入 q 退出
n
输入错误
```

```
共 274 条搜索结果，共 28 页，当前第 1 页
0: D:\Study\JAVA\project\java1\data\maildir\badeer-r\all_documents\102
1: D:\Study\JAVA\project\java1\data\maildir\badeer-r\california\4
2: D:\Study\JAVA\project\java1\data\maildir\badeer-r\discussion_threads\177
3: D:\Study\JAVA\project\java1\data\maildir\beck-s\all_documents\1812
4: D:\Study\JAVA\project\java1\data\maildir\beck-s\all_documents\1937
5: D:\Study\JAVA\project\java1\data\maildir\beck-s\all_documents\2174
6: D:\Study\JAVA\project\java1\data\maildir\beck-s\all_documents\2334
7: D:\Study\JAVA\project\java1\data\maildir\beck-s\curve_validation\3
8: D:\Study\JAVA\project\java1\data\maildir\beck-s\curve_validation\4
9: D:\Study\JAVA\project\java1\data\maildir\beck-s\curve_validation\5
输入 n 进入下一页，输入 q 退出
q
请输入检索类型
1 (标题) 2 (发件人) 3 (正文) 4 (退出)
```

```

共 274 条搜索结果, 共 28 页, 当前第 1 页
0: D:\Study\JAVA\project\java1\data\maildir\badeer-r\all_documents\102
1: D:\Study\JAVA\project\java1\data\maildir\badeer-r\california\4
2: D:\Study\JAVA\project\java1\data\maildir\badeer-r\discussion_threads\177
3: D:\Study\JAVA\project\java1\data\maildir\beck-s\all_documents\1812
4: D:\Study\JAVA\project\java1\data\maildir\beck-s\all_documents\1937
5: D:\Study\JAVA\project\java1\data\maildir\beck-s\all_documents\2174
6: D:\Study\JAVA\project\java1\data\maildir\beck-s\all_documents\2334
7: D:\Study\JAVA\project\java1\data\maildir\beck-s\curve_validation\3
8: D:\Study\JAVA\project\java1\data\maildir\beck-s\curve_validation\4
9: D:\Study\JAVA\project\java1\data\maildir\beck-s\curve_validation\5
输入 n 进入下一页, 输入 q 退出
n
共 274 条搜索结果, 共 28 页, 当前第 2 页
10: D:\Study\JAVA\project\java1\data\maildir\beck-s\curve_validation\8
11: D:\Study\JAVA\project\java1\data\maildir\beck-s\discussion_threads\1064
12: D:\Study\JAVA\project\java1\data\maildir\beck-s\discussion_threads\1255
13: D:\Study\JAVA\project\java1\data\maildir\beck-s\discussion_threads\1357
14: D:\Study\JAVA\project\java1\data\maildir\beck-s\discussion_threads\931
15: D:\Study\JAVA\project\java1\data\maildir\beck-s\var\21
16: D:\Study\JAVA\project\java1\data\maildir\buy-r\inbox\122
17: D:\Study\JAVA\project\java1\data\maildir\buy-r\inbox\131
18: D:\Study\JAVA\project\java1\data\maildir\cash-m\all_documents\212
19: D:\Study\JAVA\project\java1\data\maildir\cash-m\general_research\13
输入 1 进入上一页, 输入 n 进入下一页, 输入 q 退出

```

2、 基本情况

main 函数中第一个 while 使得每次检索后程序不退出, 进行一轮新的检索, 直到用户选择退出, 第二个 while 实现程序效果演示中的 (1), 引导用户选择检索类型。用户选择后引导用户输入检索信息。而后调用 **SearchMysqlInfo()** 函数检索, 检索结果存放在一个 Vector 中。再通过 **ShowResult()** 函数展示检索结果。

```

public static void main(String[] args) {
    while(true){
        String type;
        String target;
        while(true) {
            System.out.println("请输入检索类型\n1 (标题) 2 (发件人) 3 (正文) 4 (退出)");
            Scanner input = new Scanner(System.in);
            type = input.next();
            boolean flag = true;
            switch (type){
                case "1": type = "Subject"; break;
                case "2": type = "From"; break;
                case "3": type = "Content"; break;
                case "4": return;
                default: flag = false;
            }
            if(flag) break;
            else System.out.println("输入错误!");
        }
        System.out.println("请输入检索内容: ");
        Scanner input = new Scanner(System.in);
        target = input.next();
        Vector<String> SearchResult = SearchMysqlInfo(type, target);
        ShowResult(SearchResult);
    }
}

```


3、 SearchMysqlInfo()

通过 SQL 的查找语句进行查找，查找结果存放在一个 Vector 中并返回，核心代码为：

```
String sql = "SELECT Path FROM Email WHERE " + type + " LIKE \"%\" + target + "\"";
```

函数完整代码为：

```
static Vector<String> SearchMysqlInfo(String type, String target){
    Connection conn = null;
    Statement stmt = null;
    Vector<String> result = new Vector();
    try {
        // 注册 JDBC 驱动
        Class.forName(JDBC_DRIVER);
        // 打开链接
        conn = DriverManager.getConnection(DB_URL, USER, PASS);
        // 执行 sql 语句
        stmt = conn.createStatement();
        String sql = "SELECT Path FROM Email WHERE " + type + " LIKE \"%\" +
target + "\"";
        ResultSet rs = stmt.executeQuery(sql); // 执行 sql
        rs.last(); // 移到最后一行
        int count = rs.getRow(); // 计数
        rs.beforeFirst(); // 移到初始位置
        result = new Vector (count);

        // 展开结果集数据库
        while(rs.next()){
            result.add(rs.getString("Path"));
        }
        // 完成后关闭
        stmt.close();
        conn.close();
    } catch(Exception se) {
        se.printStackTrace();// 处理 JDBC 错误
    }
    finally{
        // 关闭资源
        try {
            if (stmt != null) stmt.close();
        } catch (SQLException ignored) {
```

```

    }// 什么都不做
    try {
        if (conn != null) conn.close();
    } catch (SQLException se) {
        se.printStackTrace();
    }
}
return result;
}

```

4、 ShowResult()

将结果分组展示，10 条为一组，引导用户进行翻页查看结果。打印结果的函数为 PrintPath()，需要注意不要溢出。

```

static void ShowResult(Vector<String> SearchResult){
    int num = SearchResult.size();
    int group = num / 10 + 1;
    int curPage = 1;

    while(true){
        System.out.println("共 " + num + " 条搜索结果，共 " + group + " 页，当前第 " + curPage + " 页");
        PrintPath(SearchResult, curPage, 10);
        if(curPage == 1) {
            System.out.println("输入 n 进入下一页，输入 q 退出");
            Scanner input = new Scanner(System.in);
            String UserInput = input.next();
            if (UserInput.equals("n")){
                curPage++;
            }
            else if (UserInput.equals("q")){
                break;
            }
            else{
                System.out.println("输入错误");
            }
        }
        else if(curPage == group){
            System.out.println("输入 1 进入上一页，输入 q 退出");
        }
    }
}

```

```

        Scanner input = new Scanner(System.in);
        String userInput = input.next();
        if (UserInput.equals("1")){
            curPage--;
        }
        else if (UserInput.equals("q")){
            break;
        }
        else{
            System.out.println("输入错误");
        }
    }
    else{
        System.out.println("输入 1 进入上一页, 输入 n 进入下一页, 输入 q 退出");

        Scanner input = new Scanner(System.in);
        String userInput = input.next();
        if(UserInput.equals("1")){
            curPage--;
        }
        else if(UserInput.equals("n")){
            curPage++;
        }
        else if (UserInput.equals("q")){
            break;
        }
        else{
            System.out.println("输入错误");
        }
    }
}

static void PrintPath(Vector<String> SearchResult, int page, int groupNum){
    int cur = (page - 1) * 10;
    for(int i = 0; i < 10; i++){
        if(cur >= SearchResult.size())
            break;
        System.out.println(cur + ": " + SearchResult.get(cur));
        cur++;
    }
}
}

```

四、遇到的问题

1、JAVA 导入 mysql 库参考：

https://blog.csdn.net/qq_38000422/article/details/79962314

2、Mysql 中存储邮件内容时，一开始使用 VARCHAR 类型，后来发现有的邮件过长无法存储，改成 TEXT，还是不行，后来改成 LONGTEXT 类型。

3、Java heap space：邮件太多，分 2-3 次录入 mysql。