

课程名称：Java应用技术 指导老师：翁恺 学生姓名：金大为

实验名称：XLSX2SQLite 实验类型：编程型 学生学号：3160103872

1. 实验目的和要求

目的：

学习Java进阶语法，了解Java的apache POI和SQLite JDBC等包的作用和使用方法，复习数据库相关知识和结构化查询语言（SQL）。使用apache POI的XSSF包实现对excel 2007+工作表（.xlsx）的读取，并将读取后的数据使用SQLite JDBC API写入SQLite数据库。

实验要求：

编写一个命令行程序，它具有三个命令行参数：

1. 数据库名
2. xlsx文件名，带.xlsx后缀
3. Excel中的页名，当这个参数省略时，取第一个页
4. 数据库表名，当这个参数省略时，用页名或当第三个参数也省略时用xlsx文件名前缀作为表名

当然第3个参数省略时不能单独带有第四个参数。

程序要打开xlsx文件，找到指定的页，在本地的SQLite数据库中建立指定名称的表。xlsx里的第一行作为表中的字段名，字段类型由xlsx数据推断得到。推断的规则为：

* 整数的为int类型；
* 浮点数为real类型；
* 其他为char类型，大小为所有行中最大的长度。

然后程序将所有的行数据导入该表，并为该表建立一个PK，以表达行号。 程序输出表的结构（SQL）和行的数量。

1. 实验内容和原理
2. 导入读取excel文件的相关API（如apache POI或jxl.jar）；
3. 使用API中的方法对excel进行处理。处理内容主要包括：表单（Sheet）的选择，第一行（关键字名称）的读取，行数和列数的读取，单元格的定位及单元格内字符串的读取。数据存入二维的List。
4. 编写方法判断字符串类型并存入List。由于JAVA excel API的type方法只能判断类型为STRING/NUMERIC/BOOLEAN/DATE等基本类型，无法实现作业要求中的判断类型是否为整数或者浮点数，因此需要自己设计方法判断某个字符串是否为整数或浮点数。
5. 导入SQLite JDBC API，连接数据库。
6. 根据输入的参数选择数据库，根据参数和type list建立表，从data list中读取数据写入表。
7. 打印建立表的SQL语句和表的总行数。
8. 主要仪器设备

LG Gram Laptop, 8GB RAM, Intel Core i5-8250U, Windows 10

IDE：IntelliJ IDEA 2018.3, JRE1.8.0

1. 操作方法和实验步骤
2. 分析实验要求，建立IntelliJ IDEA工程。
3. 导入JAVA excel API。最开始我选择了jxl.jar作为本工程的API，事实上该开源包在大量数据读取时有一定的效率优势，但是有一个被我忽视的致命缺点：无法读取excel 2007及更高版本的.xslx文件。在发现该问题后我决定先把功能实现完成，再把excel的API转换成apache POI（XSSF明确标出支持excel 2007文件）。

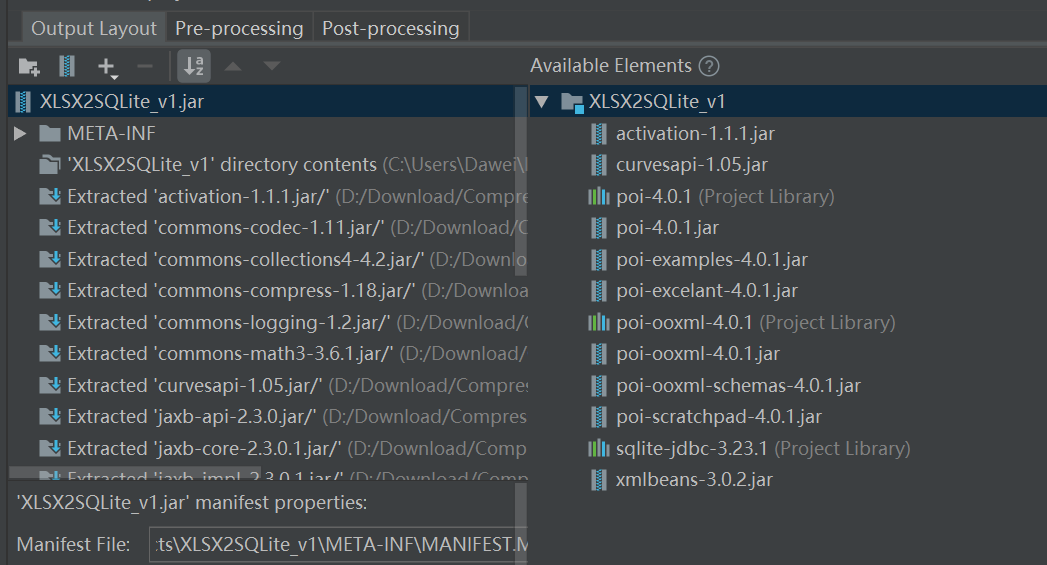


图4.1 相关API依赖包

1. 编写处理excel的部分代码。根据作业要求，程序运行需要读取2-4个参数，因此首先对args[]长度进行判断，再进行下一步操作。其中需要注意的是，由于我读取excel文件的参数是完整路径，而作业要求中有如下表述：“数据库表名……用页名或当第三个参数也省略时用xlsx文件名前缀作为表名。”因此我使用String.split方法将路径分隔开，取最后一段，再去掉后缀名。分割的标志是“\”，在split函数中要写成“\\\\”。

|  |
| --- |
| **if**(args.length >= 3) {  sheetName = args[2];  **if**(args.length >= 4) {  tableName = args[3];  }  **else** {  tableName = args[2];  }  }  **else**{  String[] fullName = fileName.split("\\\\");  **if**(fullName.length > 1) {  tableName = getFileNameNoEx(fullName[fullName.length - 1]);  }  } |

表4.1 参数处理

1. 使用以上读取的参数读取对应的工作簿和工作表，按照读取的行数和列数遍历除第一行外的所有cell的数据，以字符串形式存入列表。第一行列表作为fieldList存储关键字的名称。第二行数据用于判断数据类型。Char类型的长度方面，我使用一个全局变量maximum更新读取到的所有字符串里的最大值。需要注意的是，NUMERIC类型被POI默认处理成double，此时需要对数据进行格式化。相关代码如下：

表4.2 对数值进行格式化

|  |
| --- |
| **if**(judgeCell.getCellType() == CellType.NUMERIC) {  DecimalFormat df = **new** DecimalFormat("#.#########");  String cellText = df.format(sheet.getRow(1).getCell(i).getNumericCellValue());  **if**(isInteger(cellText)) {  typeList.add("integer");  }  **else** {  typeList.add("real");  }  } |

1. 导入SQLite JDBC API，连接数据库。使用Connection.createStatement执行建立table的语句。在建表时，按照作业要求首先插入一个ID字段作为主键，然后使用从fieldName列表中依次读取字段名称、从typeList中依次读取字段类型，拼接成Create table语句并executeUpdate。在此步骤前可以先行使用if exist语句判断是否有重名table，如有则直接drop再create。

表4.3 连接数据库并建表

|  |
| --- |
| conn = DriverManager.getConnection(url);  stat = conn.createStatement();  stat.executeUpdate("drop table if exists " + tableName + ";");  SQLStruct = "Create table " + tableName + "(" + fieldsToAdd.toString()+ ");";  stat.executeUpdate(SQLStruct); |

1. 从列表中按行、列遍历读取字符串，向表中插入数据。

图4.4 插入数据

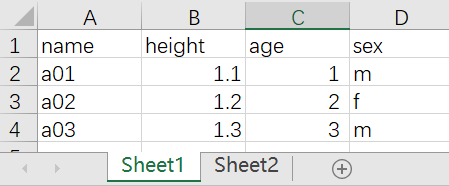
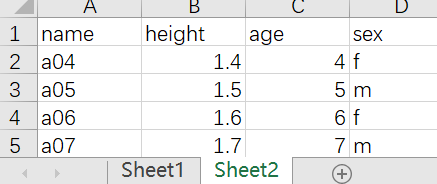
|  |
| --- |
| PreparedStatement prs = conn.prepareStatement(insertSql.toString());  **for** (int i = 1; i < outerList.size(); i++) {  prs.setInt(1, i);  **for** (int j = 0; j < innerList.size(); j++) {  prs.setString(j + 2, outerList.get(i).get(j));  }  prs.addBatch();  conn.setAutoCommit(**false**);  prs.executeBatch();  conn.setAutoCommit(**true**);  } |

1. 按照作业要求打印表结构（SQL）和行数。

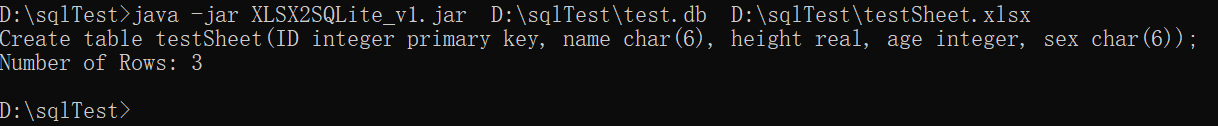
实验数据记录和处理

1. 建立测试目录，目录结构和excel内容如下图：

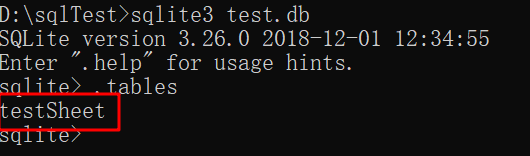


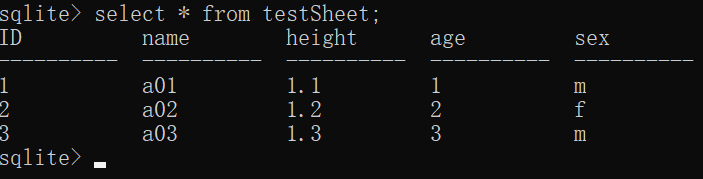
1. 只有数据库名和excel文件名作为参数。程序运行如下：



查看此时的数据库中存在的表：

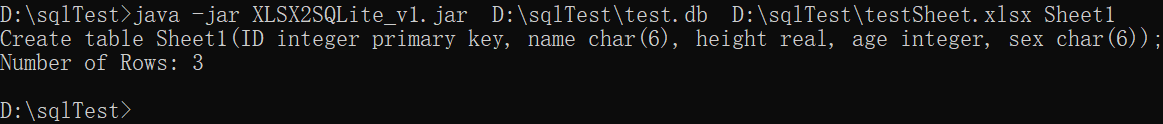


查看表中数据：

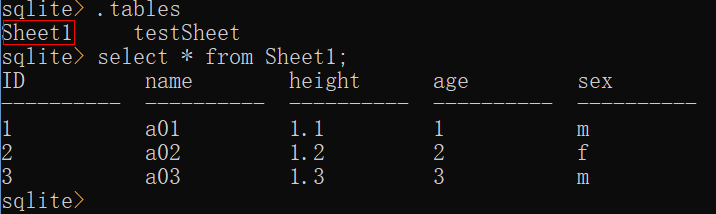


数据类型、条目数量、主键插入功能均正确。

1. 增加一个工作表名称参数。程序运行如下：

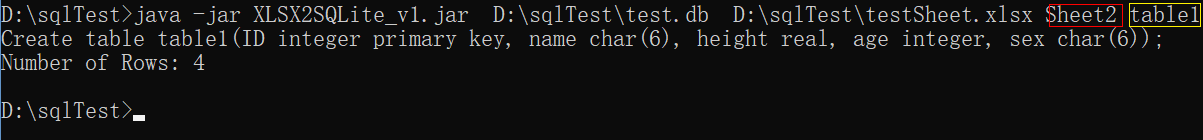


查看数据库中的表和新增表中的数据：

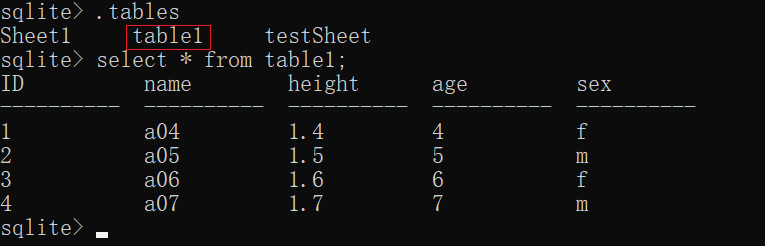


插入结果正确。

1. 增加一个数据表名称参数，也就是题目要求的四个参数。程序运行如下（此处工作表换成了Sheet2以进一步测试正确性）：

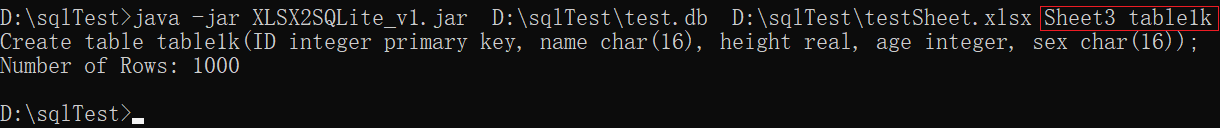


查看数据库中的表和新增表中的数据：

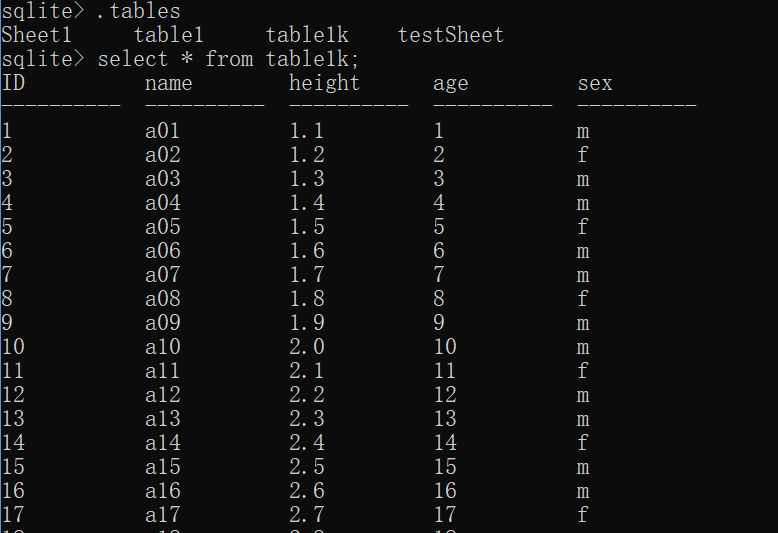


插入结果正确。

1. 压力测试：1000条数据。运行结果如下：



数据库和表更新如下：



压力测试通过。

1. 实验结果与分析

实现了作业要求，输入不同数量和内容的参数可以实现表格文件向数据库的正确导入，导入格式和内容均正确无误。

1. 讨论心得

本次作业要求和实现都比较简单，但是在实现的过程中有一些小的陷阱需要格外注意。首先是excel API的选择，对于XLSX文件类型，我们目前比较好的选择就是apache的POI，而jxl虽然轻量，却不支持新的格式，这也提醒了我在选用开源代码时要仔细查看说明文档。第二点是数据的处理，POI对于所有数值型数据都处理成double，但是作业要求我们把int区别开来，因此需要使用一些额外的方法做类型推断（比较好的方式是对字符串进行判断）。完成本次作业还需要对java依赖包的导入、SQLite语句的书写比较熟悉，这也是对相关基本操作和理论知识的一个充分复习。