XLSX2SQLITE

How to use

Open Terminal, Get to the directory that include ***VeroT2.jar*** file .

Run jar file with parameters:.

> java -jar VeroT2.jar *[path/to/db] [path/to/xlsx] [optional: xlsx/page/name] [optional: db/table/name]*

Example

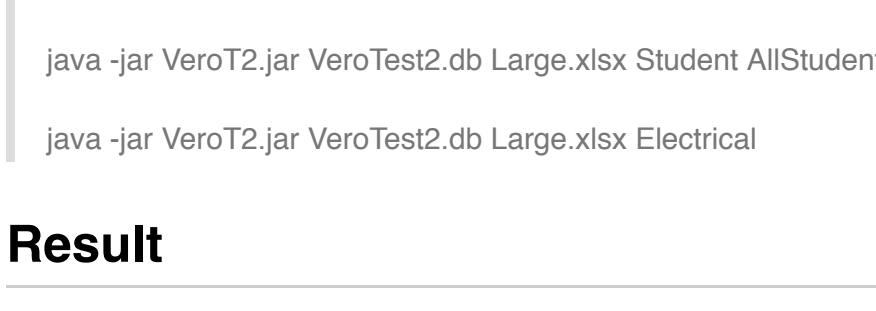
> java -jar VeroT2.jar VeroTest2.db Test.xlsx
>
> java -jar VeroT2.jar VeroTest2.db Large.xlsx Student AllStudent
>
> java -jar VeroT2.jar VeroTest2.db Large.xlsx Electrical

Result

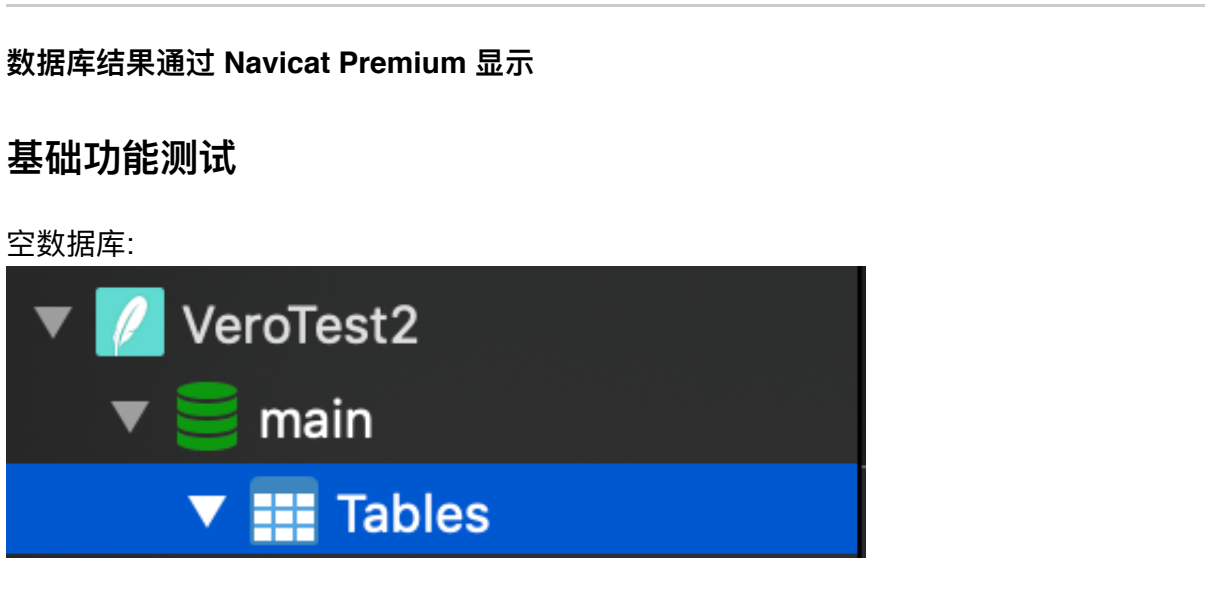
数据库结果通过 Navicat Premium 显示

基础功能测试

空数据库:

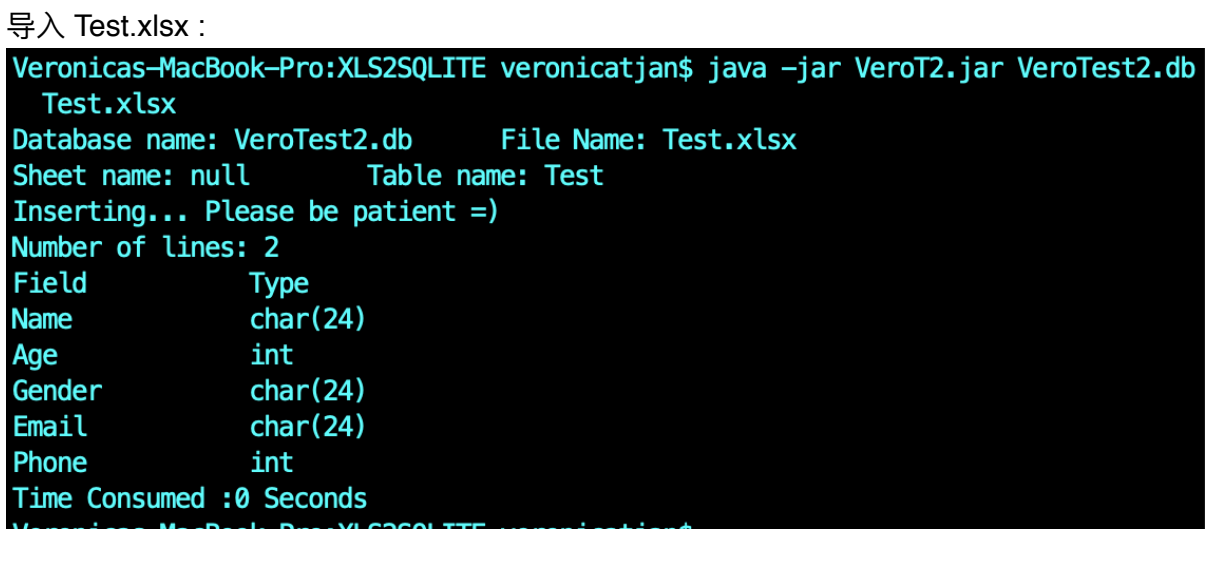


导入 Test.xlsx :



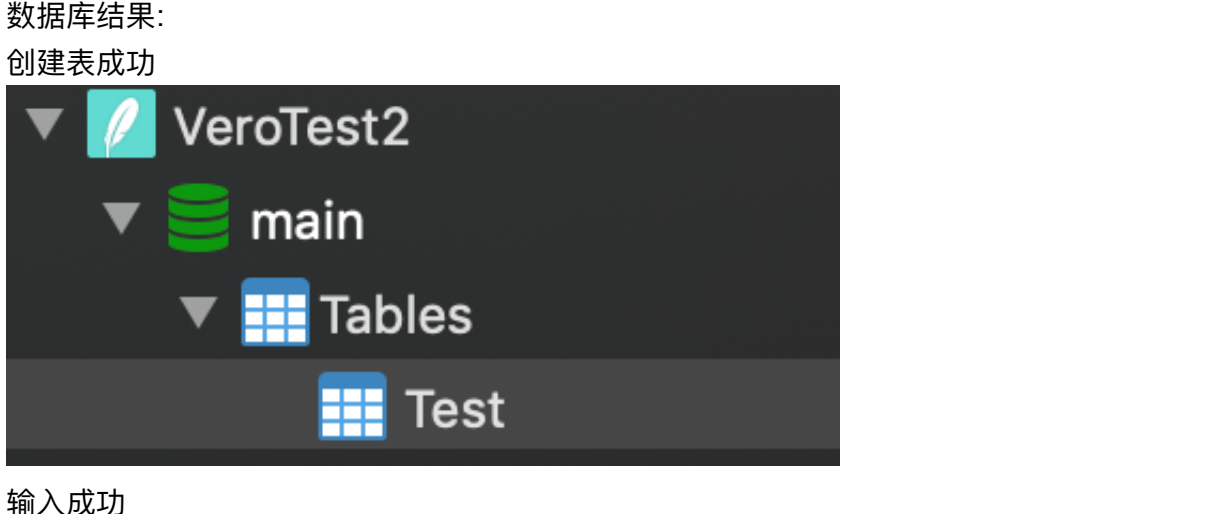
数据库结果:
创建表成功

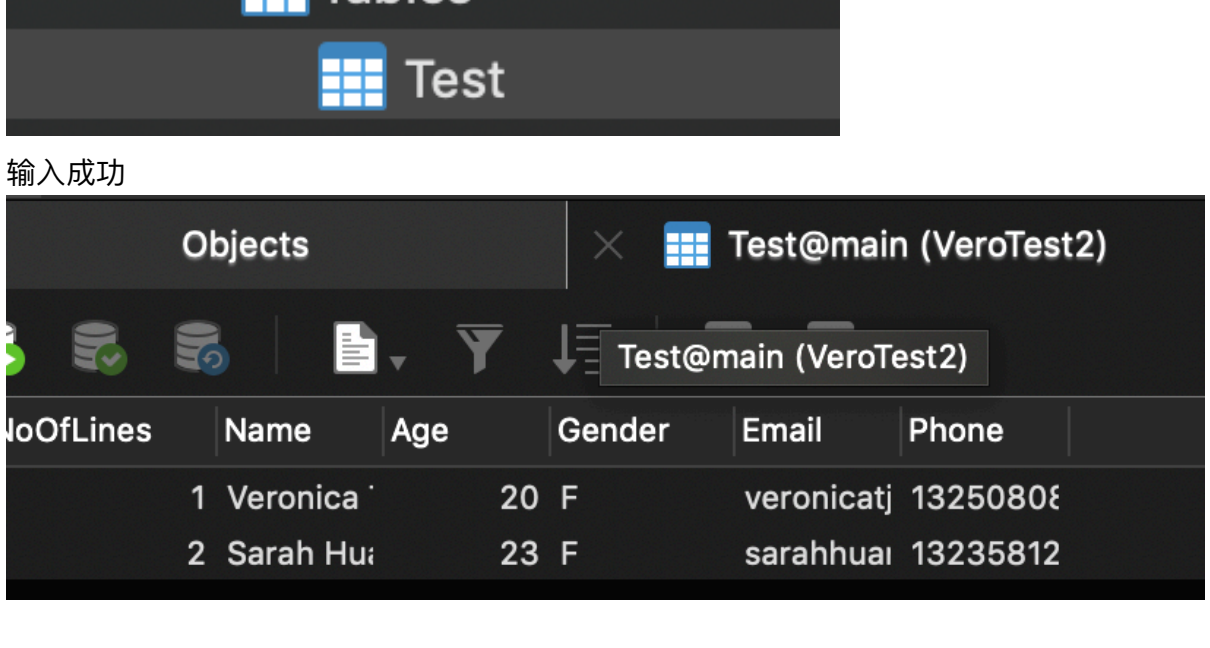
输入成功



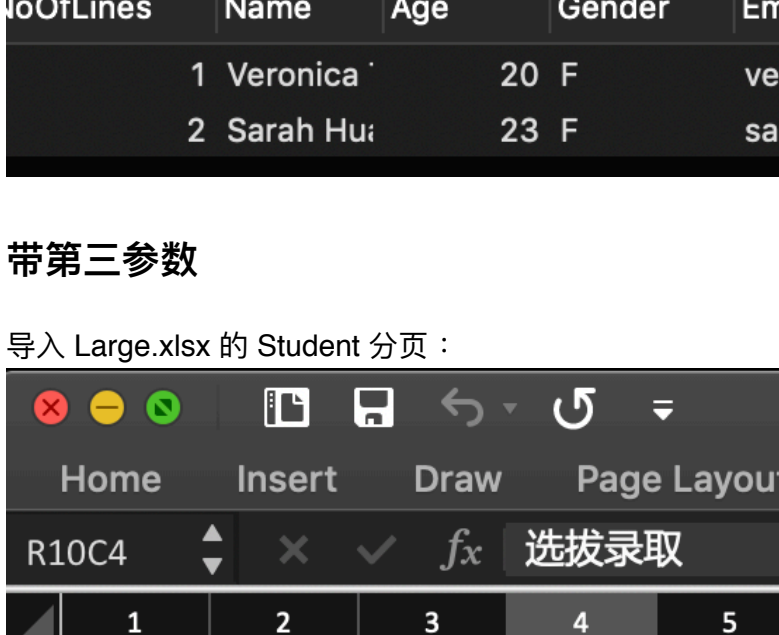
带第三参数

导入 Large.xlsx 的 Student 分页：

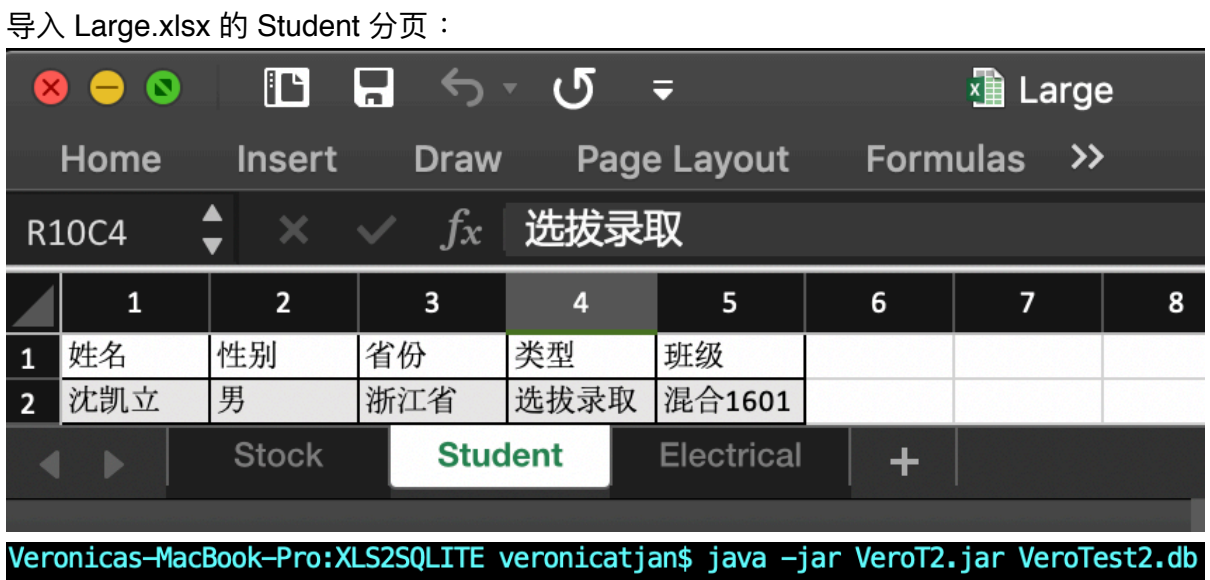


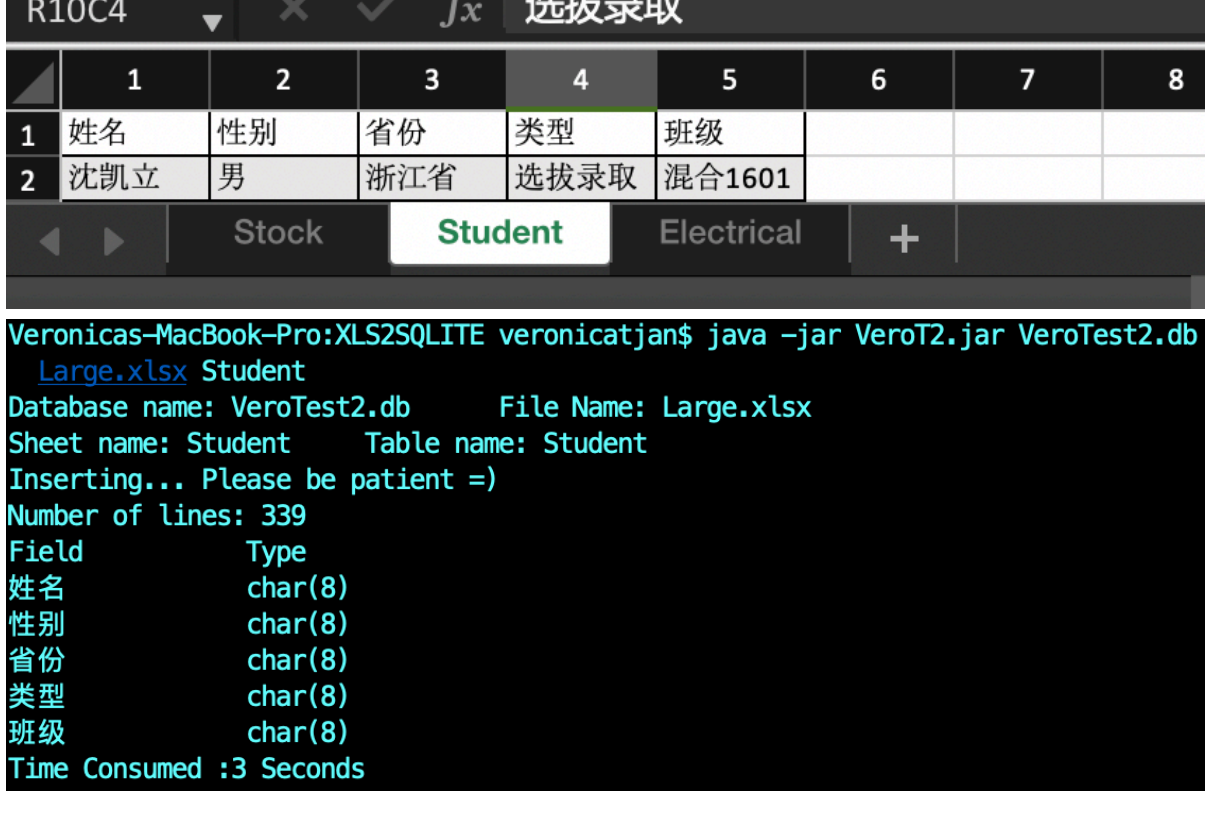


数据库结果:
创建表成功



输入成功

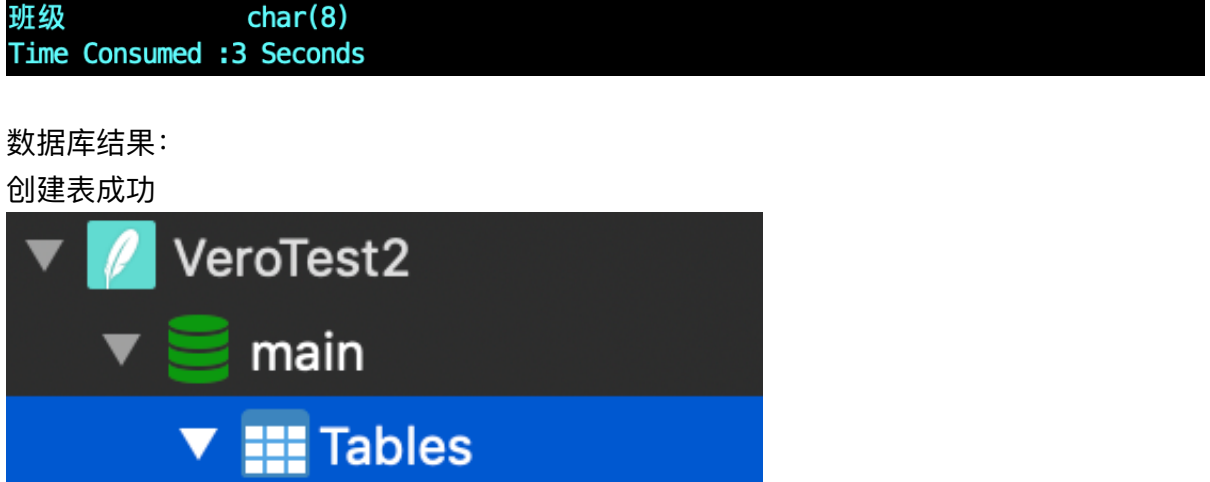


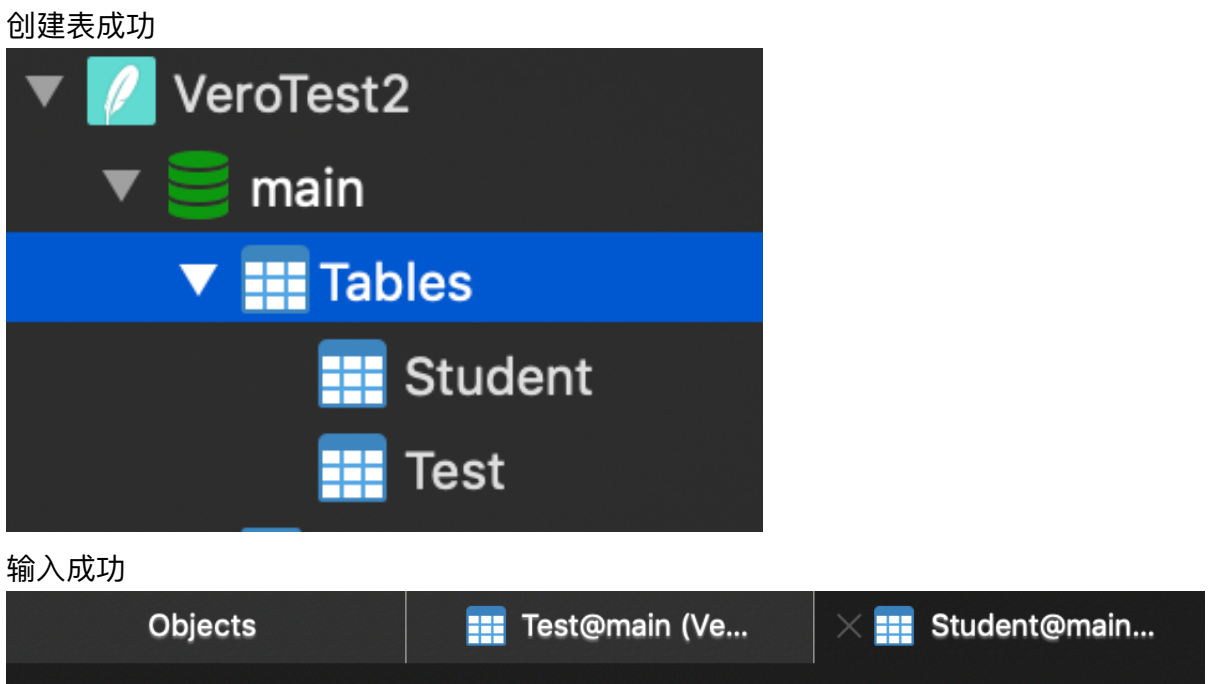


> Excel里第一行是header 所以会多一行

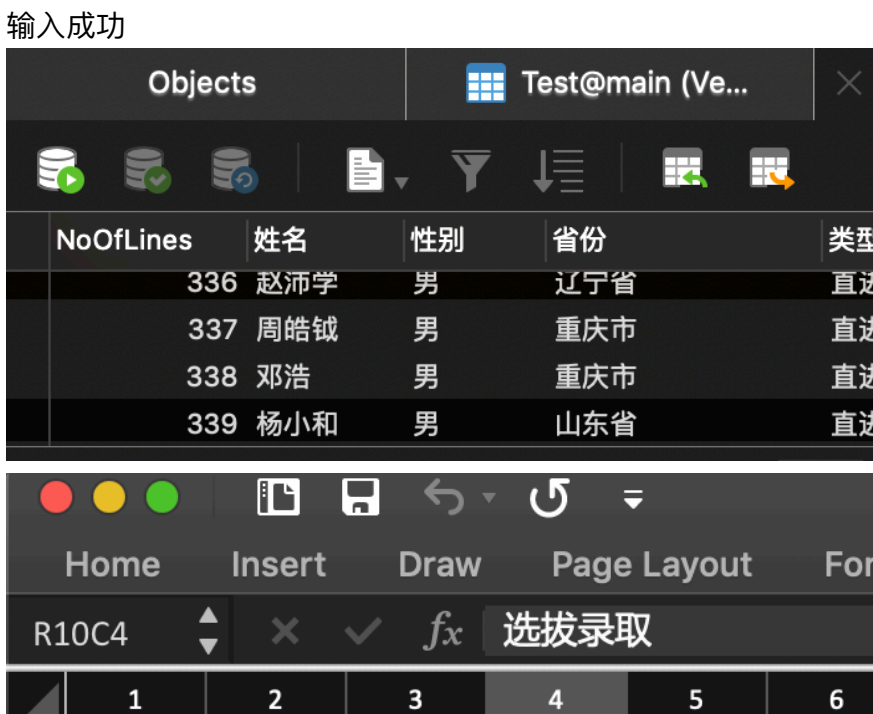
带第三第四参数

导入 Large.xlsx 的 Electrical 分页 命名表为 Elect :

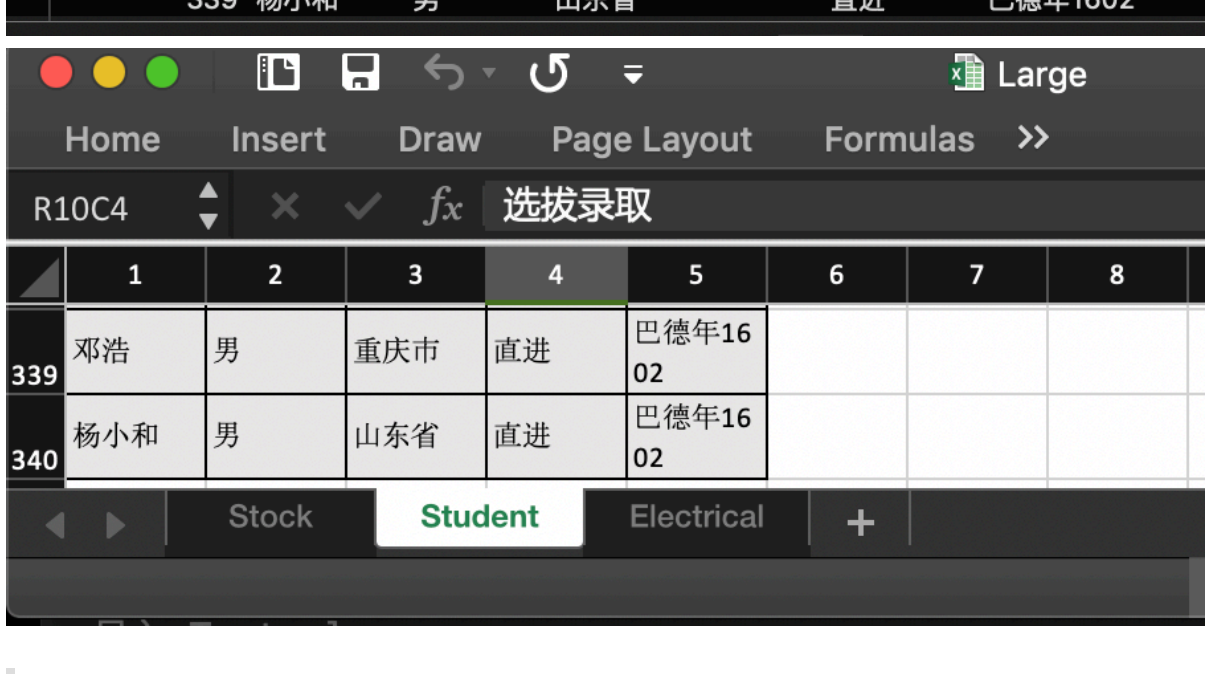


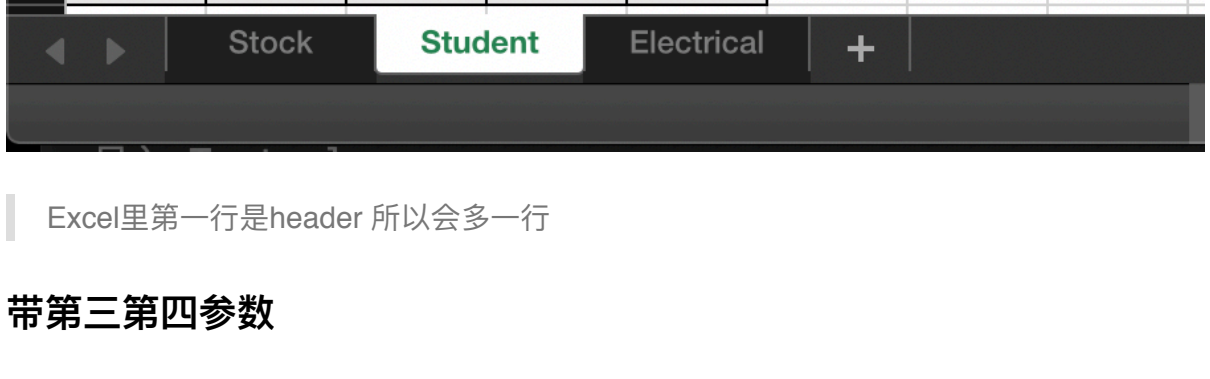


数据库结果:
创建表成功



输入成功

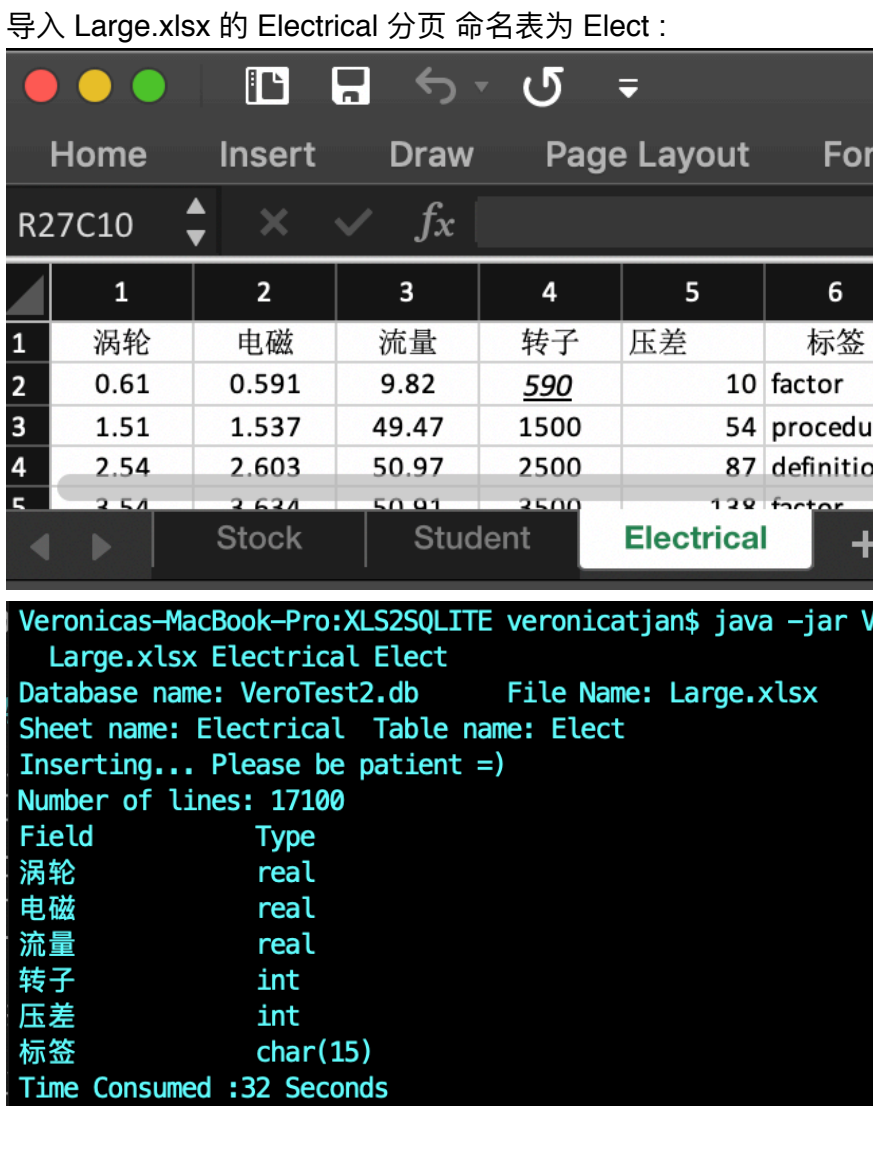




> Excel里第一行是header 所以会多一行

Implementation

Structure



Main.java

```
package com;
import java.util.Iterator;
import java.util.ArrayList;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.apache.poi.xssf.usermodel.XSSFSheet;

public class Main {
    public static void main(String[] args) throws IOException
    {
        long begintime = System.nanoTime();
        Check chk = new Check();
        //Parameters Check
        if (args.length < 2) chk.usageErr();
        if (!args[1].endsWith(".xlsx"))chk.nameErr();

        //Declarations
        DBEdit dbedit = new DBEdit();
        String DBName, FileName, SheetName, TableName;
        DBName = args[0];
        FileName = args[1];
        int LengthOfContent = 0;
        ArrayList<ArrayList<String>> ContentOfXlsx = new ArrayList<>(

        //When has at least three parameters, get SheetName
        if (args.length == 3 || args.length == 4) SheetName = args[2]
        //If there is only two parameters, get first sheet
        else SheetName = null;

        //If has 3 parameters then TableName = SheetName
        if (args.length == 3) TableName = SheetName;
        //If has 4th parameter, then get TableName
        else if (args.length == 4) TableName = args[3];
         //When there is only two parameters, TableName is FileName
        else TableName = chk.NetFileName(FileName);
        //Declare File and open xlsx file

        // Print out Database and File name
        System.out.println("Database name: " + DBName+"\t File Name:
      // Print out Sheet and table name
        System.out.println("Sheet name: "+SheetName+"\tTable name: "

        File OpenFile = new File(FileName);
        FileInputStream FileInput = new FileInputStream(OpenFile);
        XSSFWorkbook WorkBook = new XSSFWorkbook(FileInput);
        //Get Sheet from xlsx File
        XSSFSheet Sheet;

        //Get Sheet with Sheet name provided
        if (SheetName != null) Sheet = WorkBook.getSheet(SheetName);
        //Get Default page
        else Sheet = WorkBook.getSheetAt(0);
        //Sheet with Name provided is not found
        if (Sheet == null) chk.sheetnameErr();

        //Get Every Row of the sheet
        for (Row row : Sheet)
        {
            // Get every Cell of the Row
            Iterator<Cell> EveryCell = row.cellIterator();
            // Get Content of current Row
            ArrayList<String> TempContent = new ArrayList<>();
            while (EveryCell.hasNext())
            {
                Cell cell = EveryCell.next();
                TempContent.add(cell.toString());
                if (cell.toString().length() > LengthOfContent)
                    LengthOfContent = cell.toString().length();
            }
            // Add current row to ContentOfXlsx
            ContentOfXlsx.add(TempContent);
        }

        // Get current Type Of Data
        ArrayList<String> TypeOfData = new ArrayList<>();
        for (int j = 0; j < ContentOfXlsx.get(0).size(); j++)
        {
            String ColumnType = "int";
            for (int i = 1; i < ContentOfXlsx.size(); i++)
            {
                ArrayList<String> strings = ContentOfXlsx.get(i);
                // Each element of a column
                if (j >= strings.size()) strings.add("");
                String string = strings.get(j);
                if (string.equals("")) continue;
                // Integer entry
                else if (chk.isInt(string)) continue;
                // Real entry
                else if (chk.isNum(string))
                {
                    if (ColumnType.equals("int")) ColumnType = "real"
                    continue;
                }
                // String entry
                ColumnType = "char(" + LengthOfContent + ")";
            }
            TypeOfData.add(ColumnType);
        }

        // Print out line numbers
        System.out.println("Inserting... Please be patient =)");
        // Insert xlsx file into table
        dbedit.createTable(ContentOfXlsx, TypeOfData, DBName, TableNa
        dbedit.insertTable(ContentOfXlsx, TypeOfData, DBName, TableNa
        // Print out
        dbedit.showSchema(ContentOfXlsx, TypeOfData, TableName);
        // Close files
        WorkBook.close();
        FileInput.close();
        long endtime = System.nanoTime();
        long costTime = (endtime - begintime)/1000000000;
        System.out.println("Time Consumed :"+costTime+" Seconds");
    }
}
```

Check.java

```
package com;

public class Check {

    // Check is String = Number ?
    public boolean isNum(String string)
    {
        if ((string.charAt(0) > '9' || string.charAt(0) < '0')
                && string.charAt(0) != '-' && string.charAt(0) != '+'
            return false;
        for (int i = 1; i < string.length(); i++)
            if ((string.charAt(i) > '9' || string.charAt(i) < '0') &&
                return false;
        return true;
    }

    // Check is String = Integer ?
    public boolean isInt(String string)
    {
        if (!isNum(string)) return false;
        int pos = -1;
        for (int i = 0; i < string.length(); i++)
        {
            if (string.charAt(i) == '.')
            {
                pos = i;
                break;
            }
        }

        if (pos != -1)
        {
            string = string.substring(pos + 1);
            for (int i = 0; i < string.length(); i++)
            {
                if (string.charAt(i) != '0')
                    return false;
            }
        }
        return true;
    }

    // Delete .xlsx
    public String NetFileName(String string)
    {
        if (string.endsWith(".xlsx"))
            return string.replace(".xlsx", "");
        else
            return string;
    }

    public void usageErr()
    {
        System.err.println("Insert java -jar xlsx2sqlite.jar </path/t
        System.exit(-1);
    }

    public void nameErr()
    {
        System.err.println("File has to be .xlsx file");
        System.exit(-1);
    }

    public void sheetnameErr()
    {
        System.err.println("SheetName is Invalid");
        System.exit(-1);
    }

}
```

DBEdit.java

```
package com;
import java.sql.*;
import java.util.ArrayList;

class DBEdit {
    //Execute SQLite Query
    void RunQuery(String DBName, String Statement){
        String DBPath = "jdbc:sqlite:" + DBName;
        Connection conn = null;
        try
        {
            conn = DriverManager.getConnection(DBPath);
            Statement Query = conn.createStatement();
            Query.execute(Statement);
            conn.close();
        }
        catch (SQLException e)
        {
            System.out.println(e.getMessage());
        }
    }

    //Show Table Schema
    void showSchema(ArrayList<ArrayList<String>> ContentOfXlsx, Array
    {
        System.out.printf("%-12s\t%-12s", "Field", "Type");
        System.out.println();

        // Print out table
        ArrayList<String> strings = ContentOfXlsx.get(0);

        for (int i = 0; i < TypeOfData.size(); i++)
        {
            System.out.printf("%-12s\t%-12s", strings.get(i), TypeOfD
            System.out.println();
        }
    }
    void createTable(ArrayList<ArrayList<String>> ContentOfXlsx, Arra
    {
        // Delete table if exists
        String DropTableIfExist = "DROP TABLE IF EXISTS " + TableName

        // Create new table
        StringBuilder CreateTable = new StringBuilder("CREATE TABLE I
        CreateTable.append("NoOfLines int primary key,\n");

        for (int i = 0; i < TypeOfData.size(); i++)
        {
            String temp = ContentOfXlsx.get(0).get(i) + " " + TypeOfD
            if (i == TypeOfData.size() - 1)
                temp += "\n";
            else
                temp += ",\n";
            CreateTable.append(temp);
        }
        CreateTable.append(");");
        String SQLCreateTable = CreateTable.toString();
//          System.out.println("Drop : "+DropTableIfExist);
//          System.out.println("Create : "+SQLCreateTable);
        RunQuery(DBName,DropTableIfExist);
        RunQuery(DBName,SQLCreateTable);
    }

    void insertTable(ArrayList<ArrayList<String>> ContentOfXlsx, Arra
    {
        boolean IsLastLineEmpty = false;
        empty:for (int i = 1; i < ContentOfXlsx.size(); i++)
        {
            StringBuilder InsertTable = new StringBuilder();
            ArrayList<String> strings = ContentOfXlsx.get(i);
            StringBuilder temp = new StringBuilder("INSERT INTO ");
            temp.append(TableName);
            //Start Bracket for columns
            temp.append(" (");

            //Append Columns Names to the query
            temp.append("NoOfLines, ");
            ArrayList<String> names = ContentOfXlsx.get(0);
            for (int j = 0; j < names.size(); j++)
            {
                temp.append(names.get(j));
                if (j != names.size() - 1)
                    temp.append(", ");
            }

            //End Bracket for colums
            temp.append(")");
            temp.append(" VALUES (");
            //No of Lines
            temp.append(i);
            temp.append(", ");
            for (int j = 0; j < strings.size(); j++)
            {
                String string = strings.get(j);
                if (!TypeOfData.get(j).equals("int") && !(TypeOfData.
                {
                    // char
                    if(string == "")
                    {
                        IsLastLineEmpty = true ;
                        break empty; // Skip this row's content
                    }
                    temp.append("'");
                    temp.append(string);
                    temp.append("'");
                }
                else
                {
                    // int or real
                    temp.append(string);
                }
                if (j != strings.size() - 1)
                    temp.append(", ");
            }
            temp.append(");");
            InsertTable.append(temp);
            String SQLInsertTable = InsertTable.toString();
//              System.out.println("Insert : "+SQLInsertTable);
            RunQuery(DBName,SQLInsertTable);
        }
        if(IsLastLineEmpty == true) System.out.println("Number of lin
        else System.out.println("Number of lines: " + (ContentOfXlsx.
    }
}
```