数据库 lab1

毕昱阳

20302010038

目录：

一、项目结构与运行说明

项目结构：

项目使用的语言为 java，使用的数据库为 mysql。

项目根目录为文件夹 lab1_project。\src\Main.java 为主类，\src\config\config.xml 为配置文件，\src 下其他文件为实现功能用到的类。out\artifacts\lab1_project_jar\lab1_project_jar.jar 为项目打包后的 jar 包。

运行说明：

用 idea 打开 lab1_project 文件夹。项目的运行不需要修改源代码，只需要修改配置文件。

在运行之前需要修改 config.xml。在<database>标签内根据自己的数据库名还有用户名密码修改，然后在<csv_files>标签内，添加 csv 文件，每增加一个就是要新建一个<csv_file>标签，里边 url 写 csv 文件的路径，而<table_name>写这个 csv 文件对应的表名，<primary_key>写这个表的主键。这样处理的原因见。注意 csv 文件应该使用 utf-8 格式的，否则会出现中文乱码。

配置完成后，运行 Main 主类中的 main 函数，即可将 csv 文件中的数据导入数据库。

如果想要运行 jar 包，需要以下步骤。

打开 cmd，切换到项目根目录 cd　lab1_project ，

然后输入并执行这条命令： java -Dfile.encoding=utf-8 -jar out\artifacts\lab1_project_jar\lab1_project_jar.jar

二、功能实现

1.利用 jdbc 进行 java 连接数据库

这个功能的实现在 database\MyConnection 类中。jdbc 连接数据库的过程是这样的。导入 mysql 的驱动 jar 包，然后通过这条语句 Class.forName("com.mysql.cj.jdbc.Driver")来加载，然后 DriverManager 的 getConnection 方法获取连接，然后创建 Statement 对象，statement 的 execute 方法就能执行 sql 语句。MyConnection 类主要封装了建立连接和关闭连接的方法，也封装了 statement 的 execute 方法用来提交 sql 语句。

2.实现了 csv 文件的导入和解析

实现类是 file_read\MyCsvReader 和 database\String2List 。MyCsvReader 简单的读取文件，并按行存进一个 List 中。而静态类 String2List 则是输入一个 csv 文件的一行，输出根据 csv 的解析规则分开的多个数据。

3.建表以及批量插入数据

实现类是 database\Table。其构造函数是根据 csv 文件第一行，解析出表结构。同时也记录了数据库连接以及表名。建表就是根据表结构生成 CREATE TABLE IF NOT EXISTS 语句。批量插入数据就是解析多行 csv 文件并且将其拼成一句 insert 语句，然后执行。批量插入而不是每一条用一次 insert 的原因是，批量插入能极大程度减少执行时间。

对于主键，根据我自己的理解，我将 room 表的主键设置为

kdno,kcno,ccno，而 student 表的主键设置为 registno。由于这两个表之间的关系并不是很清楚，所以我暂时没有设置外键。

4.利用配置文件来设置而不是需要修改源代码

在 config\config.xml 中，利用 xml 文件来配置那些在不同环境中不同的量，例如数据库的用户名密码，还有 csv 文件的路径。

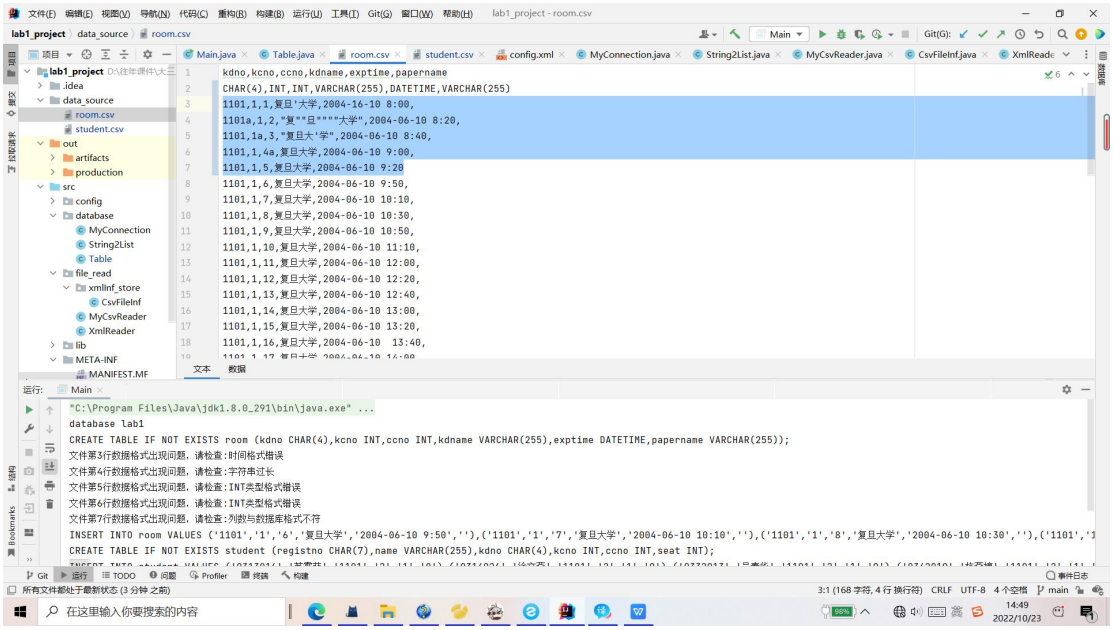
读取 xml 文件的实现主要是在 XmlReader 中。由于本 lab 中 xml 文件较短，DOM 方式和 SAX 方式的用时并没有很大差距，所以采用更容易理解的 DOM 方式。构建一个 DocumentBuilderFactory ,然后构建 DocumentBuilder，然后通过 InputStream 流读取 xml 文件，最后构建 document 对象解析 xml 文件。然后利用 getElementsByTagName，item，getChildNodes，getNodeName，getNodeValue 等方法就能获得需要的 node 中的值。

数据库连接由于只有一个数据库，所以就用了 map 来存储数据。而 csv 文件有多个，因此先用了一个类来存储每个 csv 文件在 xml 中包含的信息，然后用 List 来存储 xml 解析出来的信息。

5.安全性方面的问题

1.在建表的时候，如果第二行和第一行的列数不同，或者第二行写的并不是符合 sql 语言语法的数据类型，例如 VARCHAR(aaa)，或者是无关的字符，可能会出现问题，因此我对第二行的数据类型也进行了检查。不过由于 sql 的数据类型众多，迫于时间关系，由于本次 lab 涉及到的数据类型只有 INT,CHAR,VARCHAR,DATETIME，所以暂时没有写对于其他数据类型的支持，会报错暂未支持的数据类型。

2.对于 csv 文件，可能存在某一列列数不对或是格式不对的问题，而一旦出现语法错误，整个语句都会失效。因此在批量插入的时候，会有一个判断，假如列数和记录的表结构中的列数不符或者格式不对，那么就不会插入，同时记录下出问题的行数。这样不仅能防止列数不对的数据行插入，也不会影响正常数据的插入。实现上，就是在每个数据加入 sql 语句之前进行一次判断，用来判断格式的函数是 Table 类中的 type_check 函数。当格式不正确时会抛出异常，用来将一行 csv 文本解析成一个元组的 csv_sentence_to_sql_tuple 函数不仅判断列数是否一致，也会将 type_check 函数抛出的异常也抛出，最后在拼接 insert 语句的 insertAll 函数中 catch Exception 并加入行号信息便于查找（由于 list 下标从零开始而文件行号从一开始因此这里有 i+1 而不是 i）。



3.csv 文件的行中可能会有单引号，而 sql 中字符串是用单引号包围的，假如不做处理，那么会出现类似于“'复旦'大学'”这样的，会

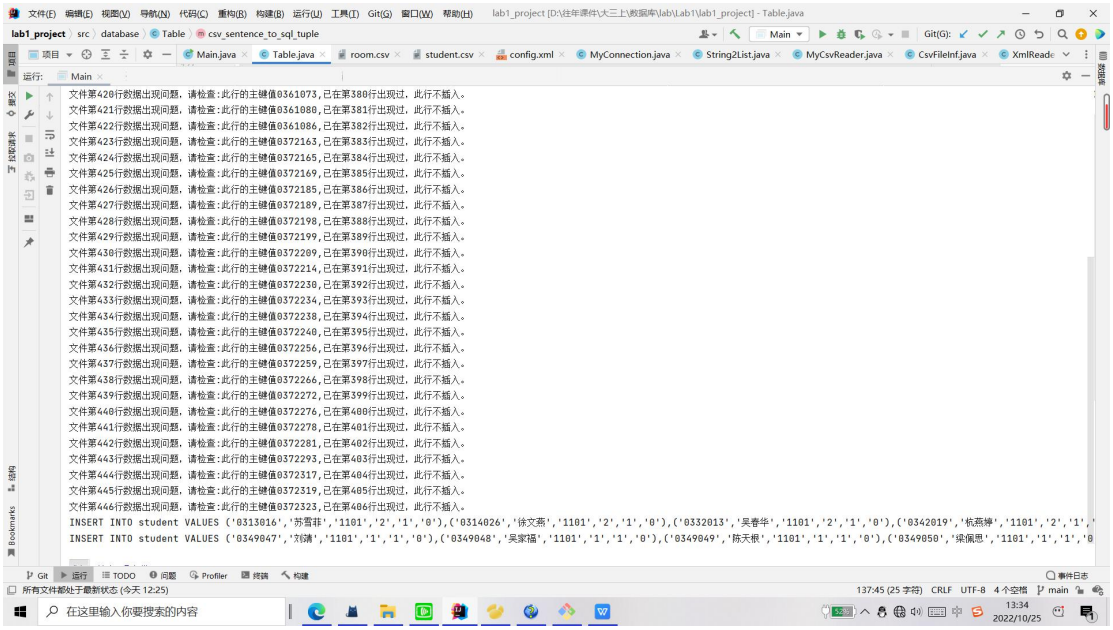
被 mysql 判定为语法错误。查阅资料得知，假如想要在数据库中存储带单引号的数据，那么写 sql 语句的时候应当连续写两个单引号，这样在数据库中就会存入一个单引号。所以我就在会插入数据库的字符串加如 sql 语句之前加了一步处理，就是将"'"（一个单引号）替换成"''"（两个单引号）。

4.对于文件内有重复主键的处理。由于我采用的是批量插入，假如文件内的重复主键不作处理，那么会因为主键重复，所有批量插入的数据都不能成功插入。因此，要在合成批量插入的语句的过程中就找出这些重复的主键，并作出处理，避免其加入到批量插入语句中导致整条语句失败。我想出了两种可行的解决方案，一种是简单的使用 insert ignore into ，这样会忽略主键重复的问题，假如有重复的就不会插入。但这种方式的缺点在于，假如 sql 语句出现了其他问题，也会忽略。同时，用户难以知道自己的文件到底哪一行出现了问题。所以我没有使用这种方法。

我的实现方式是，利用 hashmap 记录。每个元祖插入前，都将其所有主键合成的 string 作为 key，假如 hashmap 中已经存在了这个 key，说明此文件中已经有元祖的主键是存在了，所以就会抛出异常，告诉这一行的主键已经在前面出现过了。假如不存在，那么将这一行的行号存入 value。这样就能保证，相同主键的数据只有一个能插入。而且也能支持多个主键的数据库。

但这里的实现也有一定的问题。由于是在插入数据库之前进行检测，因此只能保证文件内没有主键重复，而不能保证文件内的每个元

祖的主键是否会和数据库中已有的主键重复。假如要连数据库中的元素都检测是否重复，那么 insert 语句会浪费大量时间，极大程度拖慢运行速度。而假如用 insert 语句先将所有的主键值取出来，然后比较就不需要每次都调用数据库连接的话，虽然省时间，但由于不能确定数据库有多大，假如数据库非常大的话这样实现会占用大量的空间。或许使用 insert ignore into 就能解决这些全部的问题。但 insert ignore into 的缺点是用户难以知道文件的哪一行出现了问题。我这里没有进行特殊的处理，只是当 catch 到 SQLException 的时候，给出笼统的提示。



三、问题及解决方法：

1.文件乱码

是因为 csv 文件不是用 utf8 格式的。改成 utf8 格式就好了。

2.数据 insert 速度过慢

Student 共 1885 条数据，如果写 1885 个 insert 语句的话，速度确实会非常慢。解决方法是批量 insert，INSERT INTO student VALUES

(xxx,xxx),(xxx,xxx),(xxx,xxx),(xxx,xxx)……;

3.发现数据导入的不全，只导入了 1000 条数据

这个问题有两方面原因，首先，我拼 sql 语句用的是 StringBuilder，但它有长度限制 65535，假如数据非常多的话，一条语句可能不行，因此我改成了每次只 insert1000 条，这样时间也很短但不怕超长。然而发现改了之后还是只有一千条。查阅资料得知，是因为 Navicat 的分页每页最多显示 1000 条，所以其实只要翻个页就好了。

4.student 表中每一项都额外多了个双引号

源 csv 文件中每一项也都额外多了双引号。查阅资料得知，csv 文件并没有直接按逗号分割那么简单，还有其他格式要求：

每条记录占一行，以逗号为分隔符，逗号前后的空格会被忽略。字段中包含有逗号，该字段必须用双引号括起来；字段中包含有换行符，该字段必须用双引号括起来；字段前后包含有空格，该字段必须用双引号括起来；字段中的双引号用两个双引号表示；字段中如果有双引号，该字段必须用双引号括起来。

也就是说，解析 csv 行不能简单的 split(",")，要考虑的情况更多。因此我修改了 csv 行解析的代码，具体思路如下：

一个变量 start 表示下一个要解析的字段的起始 index。也就是说它是初始在行开头，后来在逗号后边。循环条件应该是 while(start<length)，循环体内先看 sentence[start]是不是引号。

如果不是引号，就可以判定这个字段都是正常的字符，里边没有逗号也没有双引号，可以用普通的方式来判定。找到 indexOf(",",start)，

下一个逗号的位置。如果结果存在就将 substring(start,indexOf(",",start)加入 list，并且 start=indexOf+1 进入下次循环.如果结果等于-1 说明这个字段已经是最后，没有逗号了，那么就 substring(start,length).特殊的，如果逗号的 index 为 length-1，那么还需要将一个空字符串加入结果 List，因为最后一个字段是空字符串。

如果是引号，那么 start 下标就是左引号，从 start+1 找 indexof("\""),找到了也不一定是字段的结束，还要看其下一个字符是否还是双引号。如果不是说明成功找到了结束，如果找不到，那要跳过这两个引号继续往后找。还有一个要注意的是 csv 中两个连续双引号代表实际数据的一个双引号，所以还要进行一次转换。

5.只从 csv 的第一行读表结构的话缺乏数据类型和主键。

对于缺乏表结构这个问题我想出了两种解决方案。第一种方案是改变 csv 文件的结构，让它的第二行变成存储数据类型，例如第二行变成 CHAR(4),INT,INT,VARCHAR(40),DATETIME,VARCHAR(10)这种形式，然后第三行开始才是真正的数据。第二种解决方案额外创建一个文件，专门储存建表语句。由于新增加一个文件比修改原有文件的结构麻烦一些，而且 lab 要求中没有说不允许改变给定的 csv 的文件结构，所以我选择了改变 csv 文件的格式，让他第二行记录数据类型。之前向 Table 类中传输的是 csv 文件的第一行，那么改成传输第 1 和 2 行即可，数组也能很方便的记录每一列的数据结构。然后在将数据加入 insert 语句之前还可以根据数据类型来进行一下初步检查，避免出现语法错误。假如有格式错误那就和前面提到的列数不对的问题用一样

的处理方法，记录出错行数，同时这一行数据不加入 insert 语句。

但主键，一开始我也想将其加入 csv 文件第三行，但感觉不太好看，所以就将主键的设置放到了 xml 文件中。

6.由于 xml 在源代码中使用的是相对路径，导致 jar 包必须在项目文件中才能运行，无法单独拿出来运行。

xml 使用相对路径，假如是把项目源代码放到 ide 上运行，是没有任何问题的。之前用过的开源工具 jar 包都是可以单独运行的，但此程序的 jar 包依赖 xml 中的配置数据导致无法单独运行。但是读取 xml 也不可能改成绝对路径，这样还是解决不了 jar 包没法单独拿出来执行的问题，同时到了新环境运行还需要修改源码。一个应该可行的解决方案是不使用 xml 读取数据，而是设计一个交互式页面，由用户输入数据库用户密码，现场选择 csv 文件。