# 第14讲 基于Java的网络数据库程序设计

[需要2个教学单位]

**教学与实践目的：**学会基于网络数据库的网络应用程序设计技术。

## 一、概论

网络数据库程序设计就是利用网络共享(或操作)远程服务器数据库中的数据，从而实现基于网络数据库的互联网络应用。

数据库一般位于异地的服务器上，客户端应用程序用JAVA语言书写。数据库和JAVA开发环境的关系如图1所示。

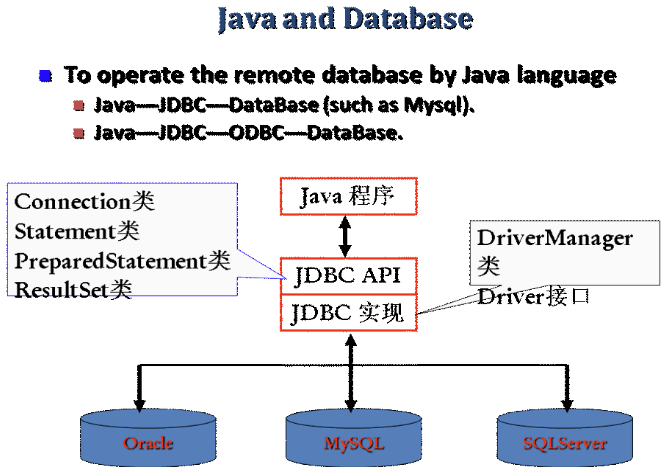


图1 JAVA与数据库

使用Java程序访问数据库时，Java代码并不是直接通过TCP连接去访问数据库，而是通过JDBC接口来访问，而JDBC接口则通过JDBC驱动来实现真正对数据库的访问。

例如，我们在Java代码中如果要访问MySQL，那么必须编写代码操作JDBC接口。注意到JDBC接口是Java标准库自带的，所以可以直接编译。而具体的JDBC驱动是由数据库厂商提供的，例如，MySQL的JDBC驱动由Oracle提供。因此，访问某个具体的数据库，我们只需要引入该厂商提供的JDBC驱动，就可以通过JDBC接口来访问，这样保证了Java程序编写的是一套数据库访问代码，却可以访问各种不同的数据库，因为他们都提供了标准的JDBC驱动。另外，类似Hibernate持久化技术是介于Java和JDBC之间的可选中间件，Java是面向对象的，大部分数据库是关系型的，Hibernate实现了它们之间的无缝连接（Java软件工程师培训教程 经济科学出版社 2015 谢建国编著）。

落实到我们的具体编程，可以看作是通过“JDBC API”和“JDBC实现”这两个中间件来实现对数据库的操作。

（1）“JDBC API”提供3个常见的类：

连接类Connection；

执行类Statement 或 PreparedStatement；

结果类ResultSet。

（2）“JDBC实现”提供2个接口：

驱动管理接口 DriverManager；

驱动接口 Driver。

## 二、程序设计第一步：构建数据库程序设计环境

（1） 阅读[附录一](#_附录一：_构建MySQL环境)，完成建库相关工作；

（2）从oc教学站点的[教学资源]=>[辅助软件资源]类别中下载[mysql-connector-java-5.1.49.jar](https://oc.gdufs.edu.cn/course/downloadResource?resourceId=279)，并将包添加到项目库中。

## 三、程序设计第二步：修改数据库配置文件解决乱码问题

在编写程序之前，这一步工作非常重要，在程序操作数据库的过程中，可能会遇到中文乱码的问题，实际上，无论什么原因出现乱码，这大多是由于数据库字符集/字符序设定的问题。

例如在MySQL连接时，出现了乱码的问题，那么基本可以确定是各个字符集/序设置不统一的原因。MySQL默认的latin1格式不支持中文，为了解决乱码问题，应该保证字符集“处处”使用统一编码，推荐的编码是utf8，为了兼容emoji表情等一些特殊字符，还可以使用utf8mb4,（完全兼容utf8）。所以在创建数据库、表、字段的时候都使用该编码。（当然，这里只是保证了数据库中的编码一致，自己程序中也应该处处使用utf8编码）

### 1. 查看当前MySQL使用的字符集

在MySQL命令行的特权模式下输入：**show variables like 'character%';**

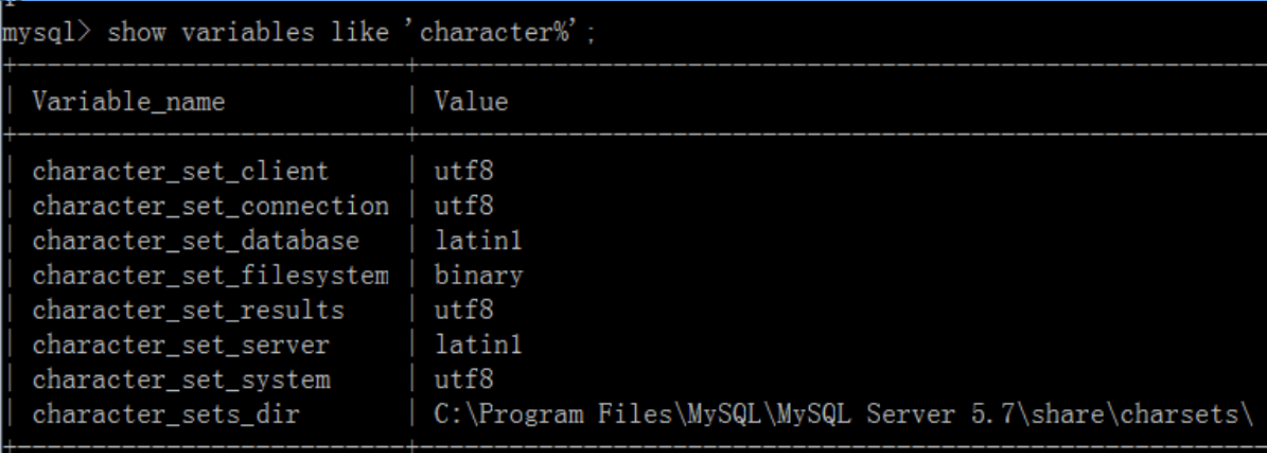


图2 未配置前使用的字符集

解析：

**character\_set\_client**：客户端请求数据的字符集

**character\_set\_connection：**客户机与服务器连接的字符集

**character\_set\_database：**默认数据库的字符集；如果没有默认数据库，就会使用 character\_set\_server指定的字符集（建议不要随意更改这个项，通过修改character\_set\_server来达到目的）

character\_set\_filesystem：把 character\_set\_client转换character\_set\_filesystem （默认为binary, 不用关注）

**character\_set\_results：**返回给客户端的字符集

**character\_set\_server**：数据库服务器的默认字符集

character\_set\_system：系统字符集，默认utf8。（用于数据库的表、列和存储在目录表中函数的名字，不用关注）

character\_sets\_dir：mysql字符集文件的保存路径（不用关注）

**产生乱码，就和以上字体加粗的几个设置项的字符集有关**，这些设置项目在不同位置发挥作用，如下图所示，一个位置不满足要求，造成编码一致，进而都可能造成乱码。

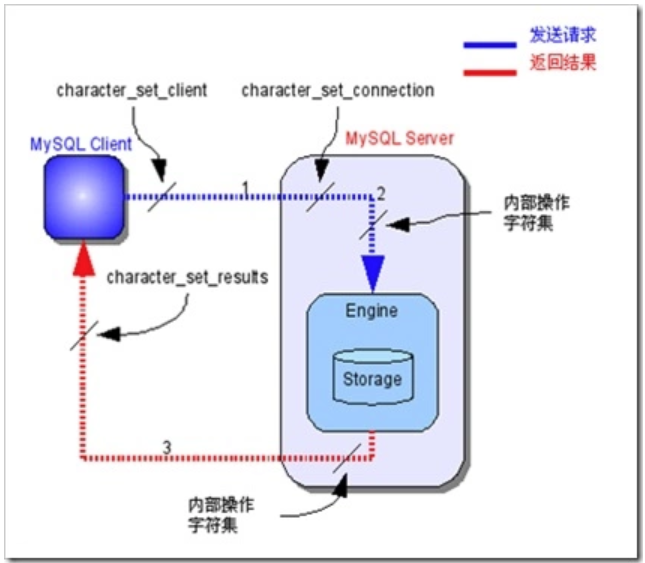


图3 字符集编码

例如图2所示，character\_set\_database和character\_set\_server编码是Latin1，这就可能出现乱码问题（注意：character\_set\_database一般不建议去主动修改，通过修改character\_set\_server可以达到目的）。

### 2. 查看当前数据库的字符序（排序规则）

除了字符集编码问题外，还要考虑一个字符序的问题。字符序不同于字符集，其用于数据库字段的相等或大小比较。修改了字符集，字符序也应该做出对应修改。

查看字符序可在特权模式下输入以下命令，结果如图4所示：

**show variables like 'collation%';**

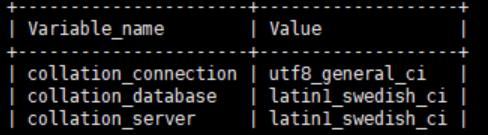


图4 未配置前的字符序

解析：

value的命名规则： 字符集名字+语言+后缀

collation\_connection ：当前连接的字符集

collation\_database ：当前日期的默认排序规则

collation\_server ：服务器的默认排序规则

后缀的含义：

ci：不区分大小写

cs：区分大小写

bin：二进制排序

### 3. 修改配置，源头解决乱码问题

从源头上解决这个问题，那就是修改MySQL默认的配置文件，把它的字符集修改成能够使用中文字符的utf8（建议改成utf8mb4，原因前面已经解释）。这样以后创建的数据库都是这种编码方式。采用的方式是修改MySQL配置文件my.cnf 或**my.ini**（windows系统下一般位于**C:\ProgramData\MySQL\MySQL Server 5.7/my.ini**， Linux一般位于/etc/my.cnf，为避免修改出错，修改前请将配置文件复制备份）。将配置文件的以下部分进行修改或添加（先搜索，存在则修改，否则进行添加。配置文件修改后要重启MySQL服务才能生效）：

[mysqld]

character-set-server=utf8mb4

collation-server=utf8mb4\_unicode\_ci

init\_connect='SET NAMES utf8mb4'

[client]

default-character-set= utf8mb4

[mysql]

default-character-set= utf8mb4

修改过程如图5所示：

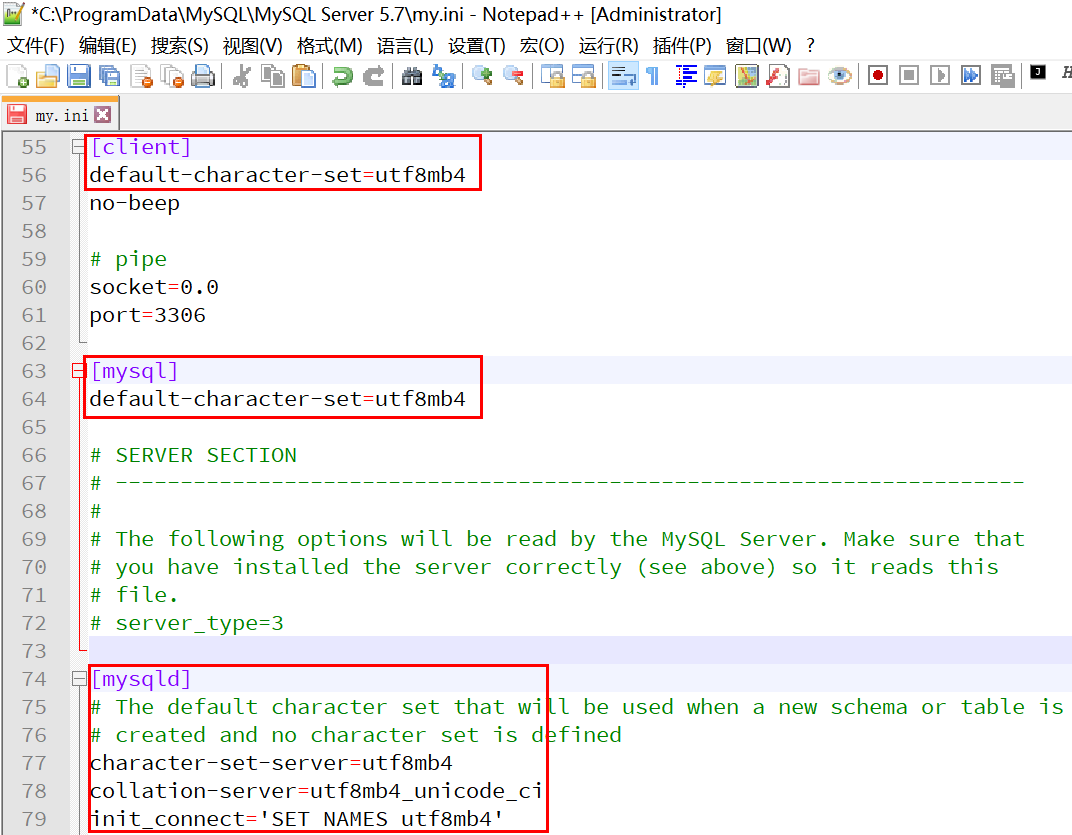


图5 修改MySQL配置文件

修改后字符集及字符序如图6所示。

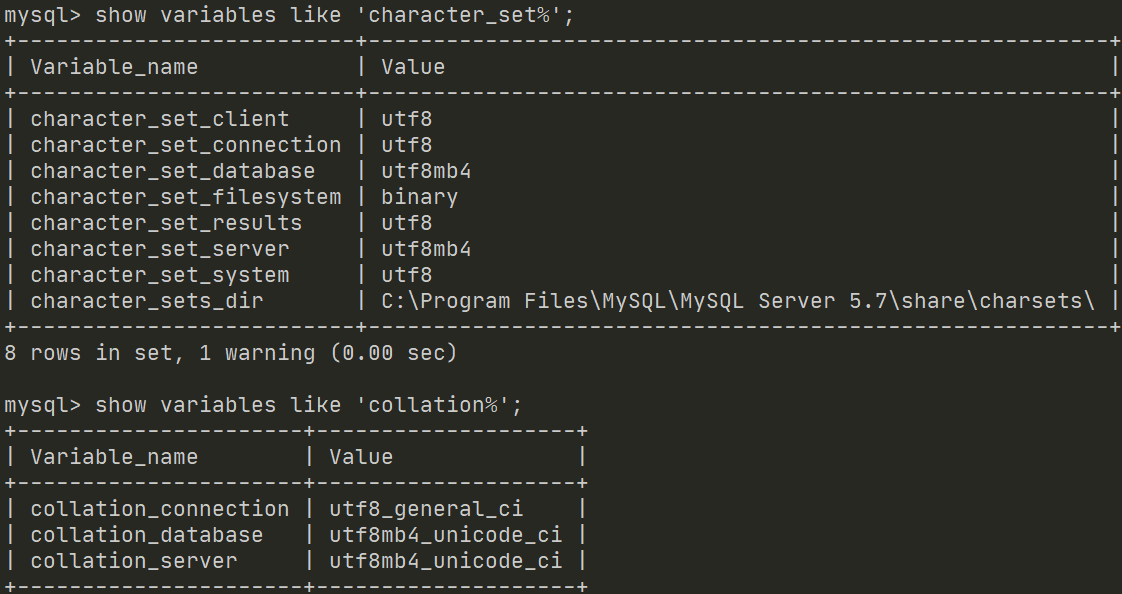


图6 修改后字符集编码及排序规则

## 三、程序设计第三步：程序操作数据库

### 1. 创建类DBOperate1

在项目中新建包chapter14，然后在此包中创建DBOperate1.java。该程序主要功能是在数据库插入一些记录，并实现浏览数据库。该程序要点包括：

（1）运用JDBC实现，为程序加载SQL数据库驱动：

*//加载MySQL驱动器，其中com.mysql.jdbc.Driver就是由下载的mysql驱动包提供*Class jdbcDriver=Class.*forName*("com.mysql.jdbc.Driver");  
  
*//注册MySQL驱动器*java.sql.DriverManager.*registerDriver*((Driver)jdbcDriver.newInstance());

（2）使用JDBC API取得数据库连接权和执行权，并进行数据库的查询操作：

*//指定数据库所在位置，先用本地地址测试，访问本地的数据库*String dbUrl = "jdbc:mysql://127.0.0.1:3306/STUDENTDB2?characterEncoding=utf8&useSSL=false";  
*//指定用户名和密码*

String dbUser="root";  
String dbPwd="本地MySQL数据库的密码";  
  
*//创建数据库连接对象*

Connection con = java.sql.DriverManager.*getConnection*(dbUrl,dbUser,dbPwd);

*//创建sql查询语句*String sql="select NO,NAME,AGE,CLASS from students where name like ? and age=?";  
  
*//创建数据库执行对象*PreparedStatement stmt = con.prepareStatement(sql);  
*//设置sql语句参数*stmt.setObject(1,"小%");  
stmt.setObject(2,23);

*//从数据库的返回集合中读出数据*ResultSet rs = stmt.executeQuery();  
  
*//循环遍历结果*while (rs.next())  
{

*//不知道字段类型的情况下，也可以用rs.getObject(…)来打印输出结果* System.*out*.print(rs.getString(1)+"\t");  
 System.*out*.print(rs.getString(2)+"\t");  
 System.*out*.print(rs.getInt(3)+"\t");  
 System.*out*.print(rs.getString(4)+"\n");  
}  
System.*out*.println("------------------------------------");

（3）在本地数据库中插入一条包含自己信息的记录，并查询：

*//设置插入记录的sql语句(如何避免重复插入学号相同的信息？)*sql = "insert into STUDENTS(NO,NAME,AGE,CLASS) values(?,?,?,?)";  
stmt = con.prepareStatement(sql);  
stmt.setObject(1,"你的学号");  
stmt.setObject(2,"你的姓名");  
stmt.setObject(3,年龄);  
stmt.setObject(4,"你的班级");  
  
stmt.executeUpdate();  
*//查询是否插入数据成功*sql = "select NO,NAME,AGE,CLASS from STUDENTS ";  
stmt = con.prepareStatement(sql);  
rs = stmt.executeQuery();  
  
*//再次循环遍历结果，看是否成功*while (rs.next())  
{  
 System.*out*.print(rs.getString(1)+"\t");  
 System.*out*.print(rs.getString(2)+"\t");  
 System.*out*.print(rs.getInt(3)+"\t");  
 System.*out*.print(rs.getString(4)+"\n");  
}

（4）释放数据库连接等资源

这一步操作非常重要，切记切记一定要保证资源的关闭，MySQL是外部资源，JVM是无法自动回收这部分资源（最好是将数据库操作的语句放在try…catch…finally中，其中释放资源语句放在finally代码块中）。

*//释放资源*if (rs != null)  
 rs.close();  
if (stmt != null)  
 stmt.close();  
if (con != null)  
 con.close();

### 2. 课堂计分

在本地数据库验证成功后，将程序适当修改，访问服务器202.116.195.71:3306上的**STUDENTDB1**数据库（用户名和密码课堂上告知），在其**STUDENTS**表中插入一条包含自己信息的记录。

服务器数据库中的STUDENTS表和大家本地创建的STUDENTS表只有一处不同：前者多了一个IP字段，该IP字段需要填入你机器的IP（即你签到记录的IP），如果插入成功，就完成本题练习，通过重新登录查分来判断是否得分。

提示：需要修改连接字符串dbUrl中的IP地址及数据库名称；由于增加了一个字段，insert语句也需要相应修改。

### 3. 常见问题

（1）在本地创建数据练习中,若出现拒绝访问本地服务器等信息，检查用户名、密码是否正确，或更改连接本机的IP地址为“localhost”或“127.0.0.1”；

（2）课堂记录不得分，注意检查连接字符串是否修改为服务器的ip地址，插入记录的IP是否为签到查分时使用的机器IP；

（3）数据库大小写敏感的问题：

默认windows 下安装的MySQL，数据库和列名是不区分大小写的，但Linux下表名会区分大小写，这种情况下，/etc/my.cnf下添加如下项：

[mysqld]

lower\_case\_table\_names=1

## 四、数据库程序设计进阶（第二周任务）

### 1. 理解JDBC API

（1）理解生命周期

执行类的生命周期由连接类创建，结果类的生命周期由执行类创建，它们各自的close() 方法可以结束自己生命周期，但必须注意顺序；

连接类生命周期 〉执行类生命周期 〉结果类生命周期；

连接类只能有一个实例化对象，连接类可以创建多个执行类实例对象，同理执行类可以创建多个结果类实例对象；

（2）结果集合类理解

一个结果集合类实例对象只保留最近一次执行的结果，前面的执行结果会被后一次执行的结果所覆盖。

### 2. 使用配置文件连接数据库

连接一个数据库，需要3个参数：数据库地址、已授权的用户名和密码，除了直接写死在源代码中，更常用的一种方式是将这些参数一般保存在一个文本文件中，程序在执行时读文本文件中的这些参数（对于程序中一些经常需要改变的内容，都可以采用这种策略，从外部的配置文件读取，修改时只需要修改外部配置文件，而不用修改程序）。

配置文件是纯文本文件，一般采用utf-8编码，使用key=value的方式编写，例如：

#配置文件使用key=value的形式，key只能为英文，不能中文

#该文件应加入到生成的类文件路径中。

JDBC\_DRIVER=com.mysql.jdbc.Driver

DB\_URL=jdbc:mysql://127.0.0.1:3306/STUDENTDB2?characterEncoding=utf8&useSSL=false

DB\_USER=student

DB\_PASSWORD=student

读取配置文件使用Java的Properties类，具体用法可见下面的例程链接：

示例程序[DBOperate2.java](https://oc.gdufs.edu.cn/download?file=DBOperate2.java)；

连接器类[ConnectionProvider.java](https://oc.gdufs.edu.cn/download?file=ConnectionProvider.java)；

以及读配置信息类[PropertyReader.java](https://oc.gdufs.edu.cn/download?file=PropertyReader.java)和配置文本文件[db.txt](https://oc.gdufs.edu.cn/download?file=db.txt)。

**常见错误：**

最常见的一个错误就是找不到配置文件。读取配置文件代码在PropertyReader.java中，采用class路径的方式读取：

InputStream in= PropertyReader.class.getResourceAsStream("/chapter14/db.txt");

建议采用绝对路径，如上面使用/charpter14/db.txt，这路径并不是在源文件中的路径，而是编译后class的路径（Java真正执行的是class文件，配置文件是相对这些文件的位置）。

如图7，这是编写代码时候，源文件的在工程项目及磁盘上的位置：

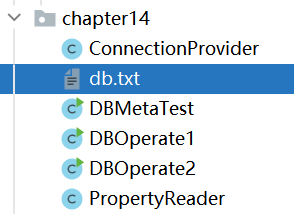
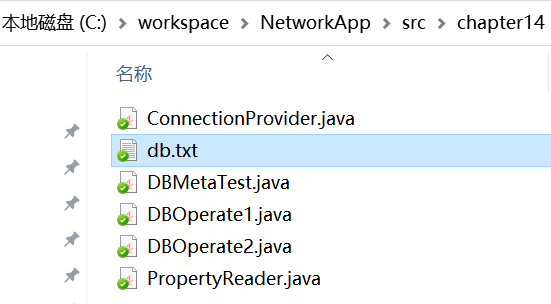
 

图7 配置文件在源代码的位置

读取的配置文件并不是这个位置，而是需要将这个配置文件拷贝到对应的类路径，如图8所示，程序读取的实际是这个位置的配置文件：

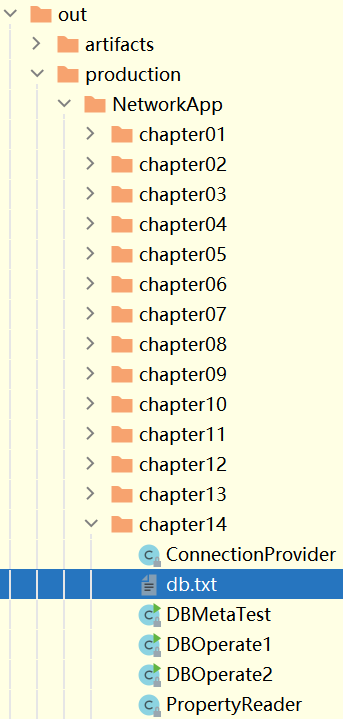
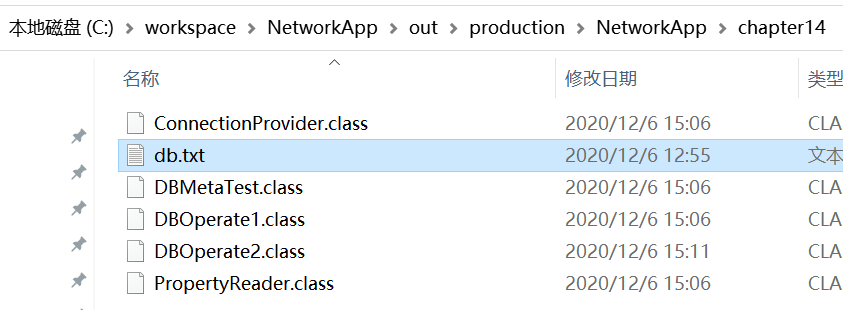
 

图8 配置文件的正确位置

可以手动将配置文件拷贝到所在类的位置，也可以在每次运行前，通过右键选择配置文件所在包，选择“rebuild .\*\*\*”，如图9所示，配置文件就会自动复制到对应的类路径（还有一种方式就是单独建一个存放配置文件的包，在ieda的Project Structure的Modules设置中，将对应的包设置为Resources，这里就不再详细结束）。

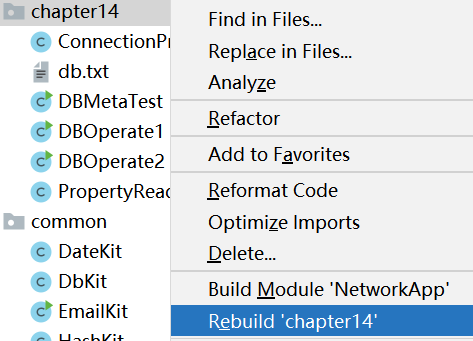


图9 设置配置文件自动复制到class路径

### 3. 课堂计分

这次需要连接202.116.195.71:3306中的mypeopledb数据库（用户名和密码课堂上告知），在该数据库中的某个表中，类似上一道题的方式插入一条自己信息的记录，通过重新登录查分，来判断是否完成本题任务。

**【提示】**

除了数据库名，你还需要知道表名和必要的字段名才能完成记录插入工作，可以通过JDBC的一些元数据操作方法，探测有哪些表名；

然后根据表名，探测这些表的字段，进而判断哪一个表才是需要插入数据的表；

如果有多个表有相同的字段，可以查询这些表中的记录，进而判断哪一个表才是最终需要插入数据的表，然后再像上一道题那样，插入一条包含自己的学号、姓名、签到IP等的记录。

以上方法可以结合断点调试，可以使用的元数据操作见下面的知识点：

（1）取得数据库连接后，如何探测表名：

DatabaseMetaData metaData = con.getMetaData();*//返回数据库的一些元信息  
/\*  
以下语句调用会返回一个ResultSet结果集，该结果集含4列，其中有一列含表名（该列名称为TABLE\_NAME，可以通过rsTables.getString("TABLE\_NAME")获得）  
可通过遍历rsTables得到包含的表名称  
\*/*ResultSet rsTables = metaData.getTables(null,null,null,new String[]{"TABLE"});

*//用于保存表名的数组列表，供之后遍历访问*  
ArrayList<String> tablesName = new ArrayList<>();  
System.*out*.println("该数据库中包含的表：");  
while (rsTables.next())  
{  
 ......  
}

（2）在表中如何探测字段名

PreparedStatement stmt = null;  
ResultSet rs = null;  
for (String tableName : tablesName)  
{  
 *//保存字段名* ArrayList<String> filedsName = new ArrayList<>();  
 String sql = "select *\** from " + tableName ;  
 stmt = con.prepareStatement(sql);  
 rs = stmt.executeQuery();  
 ResultSetMetaData fields = rs.getMetaData();*//会返回该表的字段信息* int n = fields.getColumnCount();*//有多少个字段* System.*out*.println(tableName + "字段：");  
 for(int i=1;i<=n;i++)  
 {

*//getColumnName可以获得字段名*  
 String fieldName = fields.getColumnName(i);  
 ......  
 }  
 System.*out*.println();  
 System.*out*.println(tableName + "数据：");  
 *//如果有必要，还可以循环遍历列表结果，获取有价值信息* while (rs.next())  
 {  
 ……  
 }  
 System.*out*.println("-----------------------------------");  
}

## 附录一： 构建MySQL环境

### 1. 安装MySQL免费社区版

可从OC教学站点oc教学站点的[教学资源]=>[辅助软件资源]类别中下载[mysql-installer-community-5.7.16.0.msi](https://oc.gdufs.edu.cn/course/downloadResource?resourceId=199)进行安装（生产环境中5.7是常用版本），正常来说，机房应该已经安装好了MySQL，可直接查看机房的机器中是否启动MySQL服务（从“控制面板—管理工具—服务”中查看是否存在MySQL服务，如果服务没有启动，将其启动）。

### 2. 命令行方式进入数据库管理

在程序列表中打开“MySQL command line client --Unicode”命令行界面，输入root用户的密码，就进入了特权模式，可以进行任意的数据库管理工作。（如果没有这个快捷方式，也可以控制台中进入MySQL安装位置的bin目录下，输入mysql -u root -p，回车后输入root的密码进入特权模式（不同机房密码不同，上课时由教师告知）。

### 3. 导入数据库文件

（1） 制作文本文件格式的数据库studentdb2.sql文件，并保存。内容如下所示，其中格式“#\*\*\*”属于注释语句，目的便于阅读，在studentdb2.sql中不用保留。其内容如下：

drop database if exists STUDENTDB2; #如果存在STUDENTDB2库，则删除它

create database STUDENTDB2; #创建STUDENTDB2库

use STUDENTDB2; #打开STUDENTDB2库

create table STUDENTS ( #在STUDENTDB2库中创建学生表STUDENTS，含5个字段

  ID bigint not null auto\_increment primary key,

  NO varchar(11) not null,

  NAME varchar(10) not null,

  AGE INT,

  CLASS varchar(25)

);

create table TEACHERS ( #在STUDENTDB2库中创建教师表TEACHERS，含5个字段

  ID bigint not null auto\_increment primary key,

  NO varchar(11) not null,

  NAME varchar(10) not null,

  AGE INT,

  DEPARTMENT varchar(25)

);

create table COURSES ( #在STUDENTDB2库中创建成绩表COURSES，含4个字段

   ID bigint not null auto\_increment primary key,

   NAME varchar(15),

   TIME INT,

   SCORES INT

) ;

#在学生表STUDENTS中插入2条记录

insert into STUDENTS(ID,NO,NAME,AGE,CLASS) values(1, '20141000001','小张',23, '计算机01');

insert into STUDENTS(ID,NO,NAME,AGE,CLASS) values(2, '20141000002','小m',23, '网络工程');

select \* from STUDENTS; #浏览学生表STUDENTS中所有记录

（2）将studentdb2.sql文件保存在磁盘中，例如如D盘中，然后，在MySQL命令行窗口中执行以下命令，导入studentdb2.sql，从而完成其中的建库建表等操作：

**source D:\studentdb2.sql**

注意:数据库SQL文件导入时有些平台可能要用source d:\\格式，或source d:\格式

回车后可看到执行效果如附图10所示，表明数据库及记录插入等操作成功。

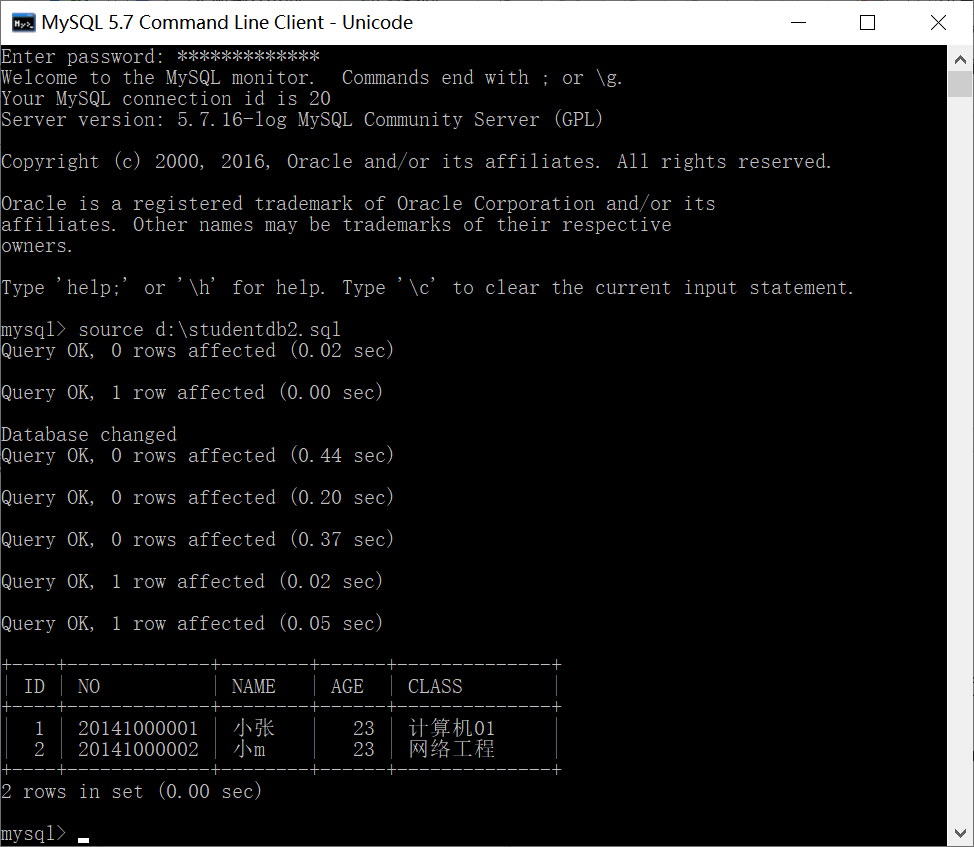


图10 数据库及表创建成功

（3）一些操作数据库的基本命令

* 查看有哪些数据库：

show databases;

* 切换到某个数据库（例如切换到studentdb2）：

use studentdb2;

* 查看数据库中的表名：

show tables;

* 查看表中字段名（例如查看students表的字段）：

show fields in students;

要退出mysql请输入exit或quit后，按回车。

其余的数据库操作命令，可自行搜索。