**数据库实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **实验人：** | **凌国明** | | **学号：** | **21307077** | **日 期：** | **2023年11月16日** |
| **院（系）：** | **计算机学院** | | | **专业（班级）：** | **计算机科学与技术** | |
| **联系方式：** |  | | |  |  | |
| **实验题目：** | | **7.2 基于JDBC的数据库应用开发实验** | | | | |

* + - 1. **实验目的**

掌握基于JDBC驱动的数据库应用开发方法。

* + - 1. **实验内容和要求**

基于JDBC驱动的数据库连接方法， 实现数据库数据操纵等应用开发常见功能。

* + - 1. **实验重点和难点**

实验重点：基于JDBC驱动的数据库连接方法数据库数据操纵功能等。

实验难点：不同的数据库应用开发工具具有不同的开发框架和模式。能够较为熟练地使用所选择的应用开发工具，是实现本实验的难点。

* + - 1. **实验工具**

MySQL、SQL Server、Navicat、NetBeans IDE 8.2（或其他开发工具）

* + - 1. **实验准备**

1. 编写实验程序的基本目标

在本实验中，以NetBeans IDE (或其他开发工具)、MySQL和SQLServer数据库为例，实现一个完整的示例程序。该程序实现把MySQL数据源中的university数据库中instructor表数据复制到SQLServer的university数据库中的T72\_instructor表中。

1. 下载JDBC驱动程序

在Java程序中要使用JDBC访问某个数据库，必须在Java项目中连接这个数据库系统所提供的JDBC驱动程序包。本实验要实现对MySQL数据库和SQLServer数据库的访问，因此要下载以下JDBC驱动程序：

(1) 下载连接 MySQL 数据库的JDBC驱动程序包（如：mysql-connector-java-8.0.30.jar）

参阅：<https://www.runoob.com/java/java-mysql-connect.html>

(2) 下载连接 SQLserver 2019 数据库的JDBC驱动程序包（如：mssql-jdbc-11.2.0.jre8.jar）

参阅：

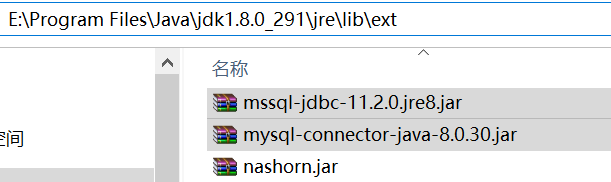
<https://blog.csdn.net/qq_45687268/article/details/125017185>

<https://blog.csdn.net/m0_68693215/article/details/124788010>

<https://blog.csdn.net/qq_46110556/article/details/116765473>

<https://learn.microsoft.com/zh-cn/sql/connect/jdbc/step-3-proof-of-concept-connecting-to-sql-using-java?view=sql-server-ver16> （注：对于参数connectionUrl，有多个例子）

* 注：为了使新建的Java程序能够直接使用下载的JDBC驱动程序包，可以将它复制到安装JDK的子目录（即：jdk\jre\lib\ext\）中，如：



1. JDBC程序的基本结构

(1)基于JDBC驱动的数据库连接方法

/\*step 1 加载驱动程序：Class.forName(driverName); \*/

/\* step 2 连接数据库 ：Connection con = DriverManager.getConnection(dbURL, userName, userPwd); \*/

(2)基于JDBC驱动的数据库数据操纵方法

/\* step 3 创建Statement/PreparedStatement对象，用来执行sql语句

/\* step 3.1 Statement stmt = con.createStatement();\*/

/\* step 3.2 PreparedStatement ps=prepareStatement(sql); \*/

/\* step 4 给占位符赋值\*/

/\* step 5 执行sql语句\*/

/\* step 5.1接收结果集：ResultSet rs = stmt.executeQuery(sqlStr); \*/

/\* step 5.2更新操作：ps.executeUpdate();\*/

/\* step 6 处理结果:遍历结果集\*/

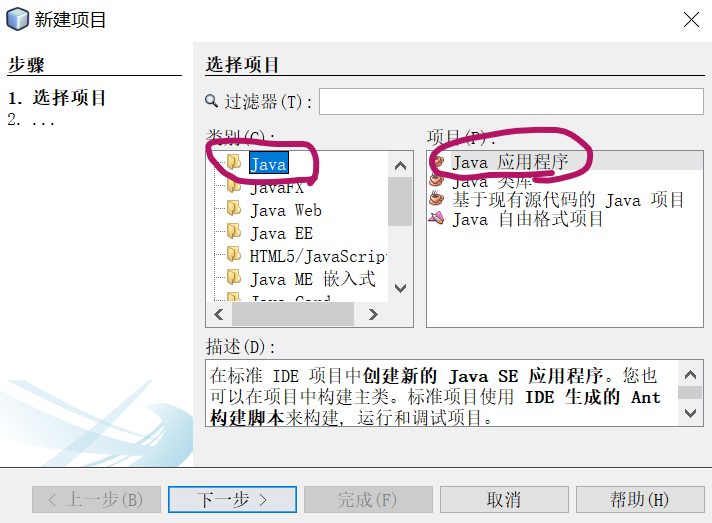
(3) 中断基于JDBC驱动的数据库连接

/\* step 7完成后关闭各个对象 \*/

* + - 1. **实验过程**

1. 编写程序T72A, 显示MySQL数据源中的university数据库中instructor表中的所有记录。

**Step 1：创建项目**T72A，如图：



**Step 2：在程序文件T72A.java**中输入以下代码：

package t72a;

import java.sql.\*;

public class T72A {

static final String JDBC\_DRIVER = "com.mysql.cj.jdbc.Driver";

static final String DB\_URL = "jdbc:mysql://localhost:3306/university?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC";

// 数据库的用户名与密码，需要根据自己的设置

static final String USER = "root";

static final String PASS = "123456";

public static void main(String[] args) {

Connection conn = null;

Statement stmt = null;

try{

// step 1 加载驱动程序：Class.forName(driverName);

Class.forName(JDBC\_DRIVER);

// step 2 获得数据库连接 ：Connection con = DriverManager.getConnection(dbURL, userName, userPwd);

System.out.println("连接数据库..."+DB\_URL);

conn = DriverManager.getConnection(DB\_URL,USER,PASS);

ResultSet rs;

// step 3 创建Statement/PreparedStatement对象，用来执行sql语句

// step 3.1 创建Statement： Statement stmt = con.createStatement();

// step 3.2 创建PreparedStatement对象

stmt = conn.createStatement();

//step 4 给占位符赋值

//step 5 执行sql语句(接收结果集)：ResultSet rs = stmt.executeQuery(sqlStr);

String sql;

sql = "select id,name,dept\_name,salary from instructor";

rs = stmt.executeQuery(sql);

// step 6 处理结果:遍历结果集

System.out.print("id\tname\tdept\_name\tsalary\n");

while(rs.next()){

// 通过字段检索

String inst\_id = rs.getString("id");

String inst\_name = rs.getString("name");

String dept\_name = rs.getString("dept\_name");

Float salary = rs.getFloat("salary");

// 输出数据

System.out.print(inst\_id);

System.out.print("\t" + inst\_name);

System.out.print("\t" + dept\_name);

System.out.print("\t" + salary);

System.out.print("\n");

}

// step 7 完成后关闭各个对象

rs.close();

stmt.close();

conn.close();

}catch(SQLException se){

// 处理 JDBC 错误

se.printStackTrace();

}catch(Exception e){

// 处理 Class.forName 错误

e.printStackTrace();

}finally{

// 关闭资源

try{

if(stmt!=null) stmt.close();

}catch(SQLException se2){

}// 什么都不做

try{

if(conn!=null) conn.close();

}catch(SQLException se){

se.printStackTrace();

}

}

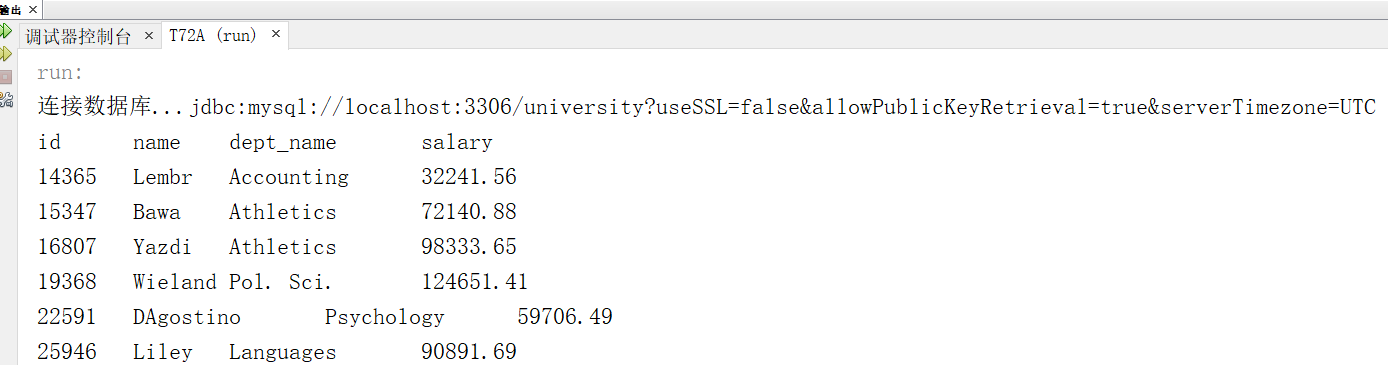
System.out.println("Goodbye!");

}

}

**Step 3：运行将显示**

验证截图如下：

****

1. 编写程序T72B, 根据程序T72A设计函数   
   public static void showTableByJDBC(String JDBC\_DRIVER,String DB\_URL,String USER,String PASS);   
   该函数显示由JDBC\_DRIVER加载的数据源DB\_URL中的university数据库中instructor表中的所有记录。

**Step 1：创建项目**T72B

**Step 2：在程序文件 T72B.java** 中输入以下代码：

package t72b;

import java.sql.\*;

public class T72B {

public static void main(String[] args) {

showTableByJDBC("com.mysql.cj.jdbc.Driver"

,"jdbc:mysql://localhost:3306/university?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC"

,"root","123456");

showTableByJDBC("com.microsoft.sqlserver.jdbc.SQLServerDriver"

,"jdbc:sqlserver://localhost:1433;DatabaseName=university;"

+ "encrypt=true;"

+ "trustServerCertificate=true;"

+ "loginTimeout=30;"

,"sa","123456");

}

public static void showTableByJDBC(String JDBC\_DRIVER,String DB\_URL,String USER,String PASS) {

Connection conn = null;

Statement stmt = null;

try{

// step 1 加载驱动程序：Class.forName(driverName);

Class.forName(JDBC\_DRIVER);

// step 2 获得数据库连接 ：Connection con = DriverManager.getConnection(dbURL, userName, userPwd);

System.out.println("连接数据库..."+DB\_URL);

conn = DriverManager.getConnection(DB\_URL,USER,PASS);

ResultSet rs;

// step 3 创建Statement/PreparedStatement对象，用来执行sql语句

// step 3.1 创建Statement： Statement stmt = con.createStatement();

// step 3.2 创建PreparedStatement对象

stmt = conn.createStatement();

//step 4 给占位符赋值

//step 5 执行sql语句(接收结果集)：ResultSet rs = stmt.executeQuery(sqlStr);

String sql;

sql = "select id,name,dept\_name,salary from instructor";

rs = stmt.executeQuery(sql);

// step 6 处理结果:遍历结果集

System.out.print("id\tname\tdept\_name\tsalary\n");

while(rs.next()){

// 通过字段检索

String inst\_id = rs.getString("id");

String inst\_name = rs.getString("name");

String dept\_name = rs.getString("dept\_name");

Float salary = rs.getFloat("salary");

// 输出数据

System.out.print(inst\_id);

System.out.print("\t" + inst\_name);

System.out.print("\t" + dept\_name);

System.out.print("\t" + salary);

System.out.print("\n");

}

// step 7 完成后关闭各个对象

rs.close();

stmt.close();

conn.close();

}catch(SQLException se){

// 处理 JDBC 错误

se.printStackTrace();

}catch(Exception e){

// 处理 Class.forName 错误

e.printStackTrace();

}finally{

// 关闭资源

try{

if(stmt!=null) stmt.close();

}catch(SQLException se2){

}// 什么都不做

try{

if(conn!=null) conn.close();

}catch(SQLException se){

se.printStackTrace();

}

}

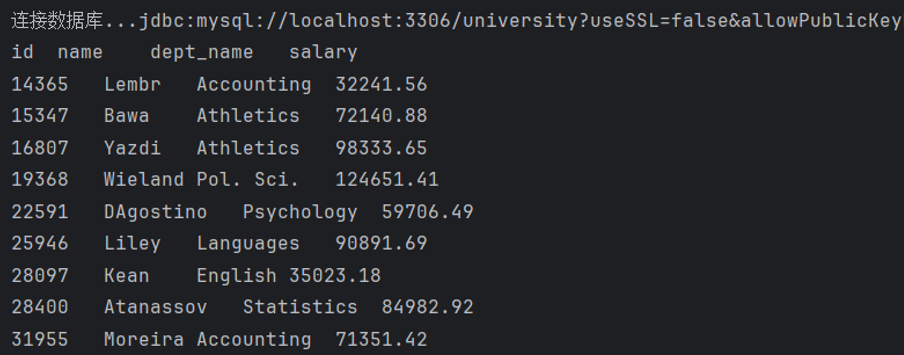
System.out.println("Goodbye!");

}

}

**Step 3：运行将显示**

验证截图如下：



1. 编写程序T72C,设计函数  
   public static void copyTableByJDBC(String src\_JDBC\_DRIVER,String src\_DB\_URL,String src\_USER,String src\_PASS,  
   String target\_JDBC\_DRIVER,String target\_DB\_URL,String target\_USER,String target\_PASS )  
   该函数功能是：
   * 先在目标数据源target\_DB\_URL创建T72\_instructor表（注：与instructor表结构一样）中，
   * 然后将源数据源src\_DB\_URL中instructor表中的所有记录复制到目标数据源target\_dsn中的T72\_instructor表中。

**Step 1：创建项目**T72C

**Step 2：在程序文件 T72C.java** 中输入以下代码：

package t72c;

import java.sql.\*;

public class T72C {

public static void main(String[] args) {

copyTableByJDBC("com.mysql.cj.jdbc.Driver"

,"jdbc:mysql://localhost:3306/university?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC"

,"root","123456",

"com.microsoft.sqlserver.jdbc.SQLServerDriver"

,"jdbc:sqlserver://localhost:1433;DatabaseName=university;"

+ "encrypt=true;"

+ "trustServerCertificate=true;"

+ "loginTimeout=30;"

,"sa","123456");

copyTableByJDBC("com.microsoft.sqlserver.jdbc.SQLServerDriver"

,"jdbc:sqlserver://localhost:1433;DatabaseName=university;"

+ "encrypt=true;"

+ "trustServerCertificate=true;"

+ "loginTimeout=30;"

,"sa","123456"

,"com.mysql.cj.jdbc.Driver"

,"jdbc:mysql://localhost:3306/university?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC"

,"root","123456"

);

}

public static void copyTableByJDBC(String src\_JDBC\_DRIVER,String src\_DB\_URL,String src\_USER,String src\_PASS,

String target\_JDBC\_DRIVER,String target\_DB\_URL,String target\_USER,String target\_PASS )

{

Connection src\_conn = null;

Statement src\_stmt = null;

Connection target\_conn = null;

PreparedStatement target\_ps=null;

try{

System.out.println("连接数据库..."+src\_DB\_URL);

System.out.println("连接数据库..."+target\_DB\_URL);

// step 1 加载驱动程序：Class.forName(driverName);

Class.forName(src\_JDBC\_DRIVER);

Class.forName(target\_JDBC\_DRIVER);

// step 2 连接数据库 ：Connection con = DriverManager.getConnection(dbURL, userName, userPwd);

src\_conn = DriverManager.getConnection(src\_DB\_URL,src\_USER,src\_PASS);

target\_conn = DriverManager.getConnection(target\_DB\_URL,target\_USER,target\_PASS);

ResultSet src\_rs;

// step 3 创建Statement/PreparedStatement对象，用来执行sql语句

// step 3.1 创建Statement： Statement stmt = con.createStatement();

// step 3.2 创建PreparedStatement对象: PreparedStatement ps=prepareStatement(sql);

src\_stmt = src\_conn.createStatement();

target\_ps = target\_conn.prepareStatement("create table T72\_instructor (ID varchar(5), name varchar(20) not null, dept\_name varchar(20), salary numeric(8,2) )");

target\_ps.executeUpdate();

target\_ps = target\_conn.prepareStatement("INSERT INTO T72\_instructor(ID,name,dept\_name, salary) VALUES(?, ?, ?, ?)");

//step 5 执行sql语句(接收结果集)：ResultSet rs = stmt.executeQuery(sqlStr);

String select\_sql;

select\_sql = "select id,name,dept\_name,salary from instructor";

src\_rs = src\_stmt.executeQuery(select\_sql);

// step 6 处理结果:遍历结果集

System.out.print("id\tname\tdept\_name\tsalary\n");

while(src\_rs.next()){

// 通过字段检索

String inst\_id = src\_rs.getString("id");

String inst\_name = src\_rs.getString("name");

String dept\_name = src\_rs.getString("dept\_name");

Float salary = src\_rs.getFloat("salary");

// 输出数据

System.out.print(inst\_id);

System.out.print("\t" + inst\_name);

System.out.print("\t" + dept\_name);

System.out.print("\t" + salary);

System.out.print("\n");

//step 4 给占位符赋值

target\_ps.setString(1, inst\_id); //给第1个占位符赋值(ID)

target\_ps.setString(2, inst\_name);//给第2个占位符赋值(name)

target\_ps.setString(3, dept\_name);//给第3个占位符赋值(dept\_name)

target\_ps.setFloat(4, salary); //给第4个占位符赋值(salary)

//step 5.2 执行sql语句(更新操作)：ps.executeUpdate();

target\_ps.executeUpdate();

}

// step 7 完成后关闭各个对象

src\_rs.close();

src\_stmt.close();

src\_conn.close();

target\_ps.close();

target\_conn.close();

}catch(SQLException se){

// 处理 JDBC 错误

se.printStackTrace();

}catch(Exception e){

e.printStackTrace();

}

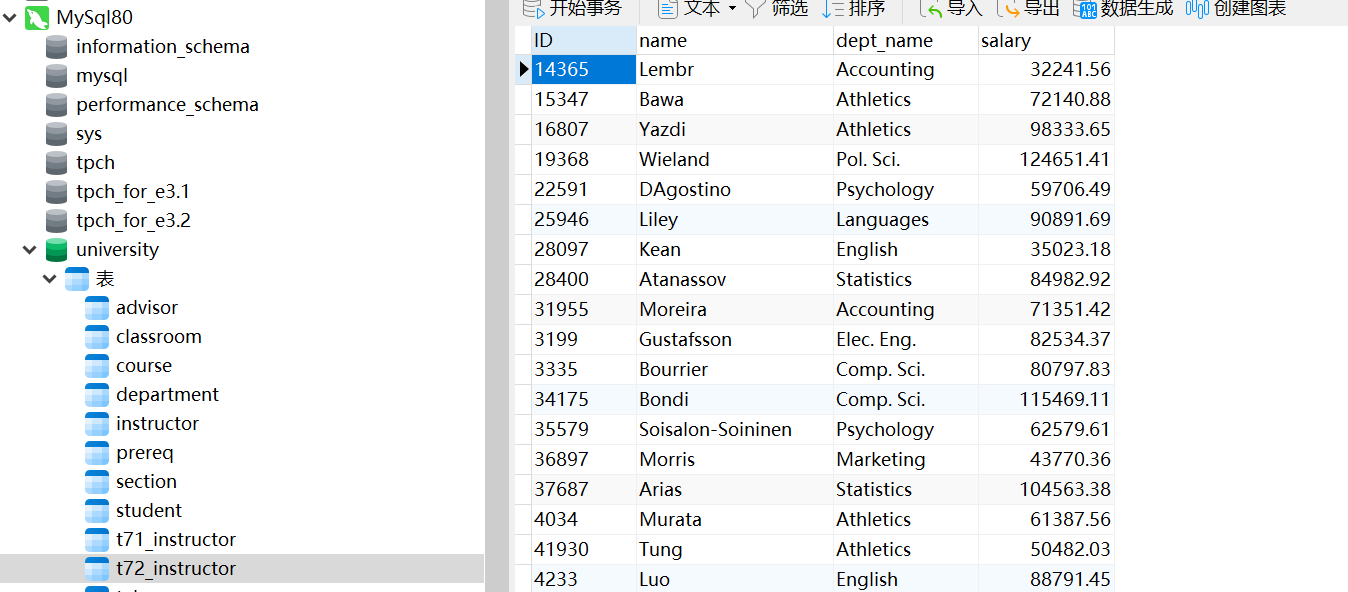
System.out.println("Goodbye!");

}

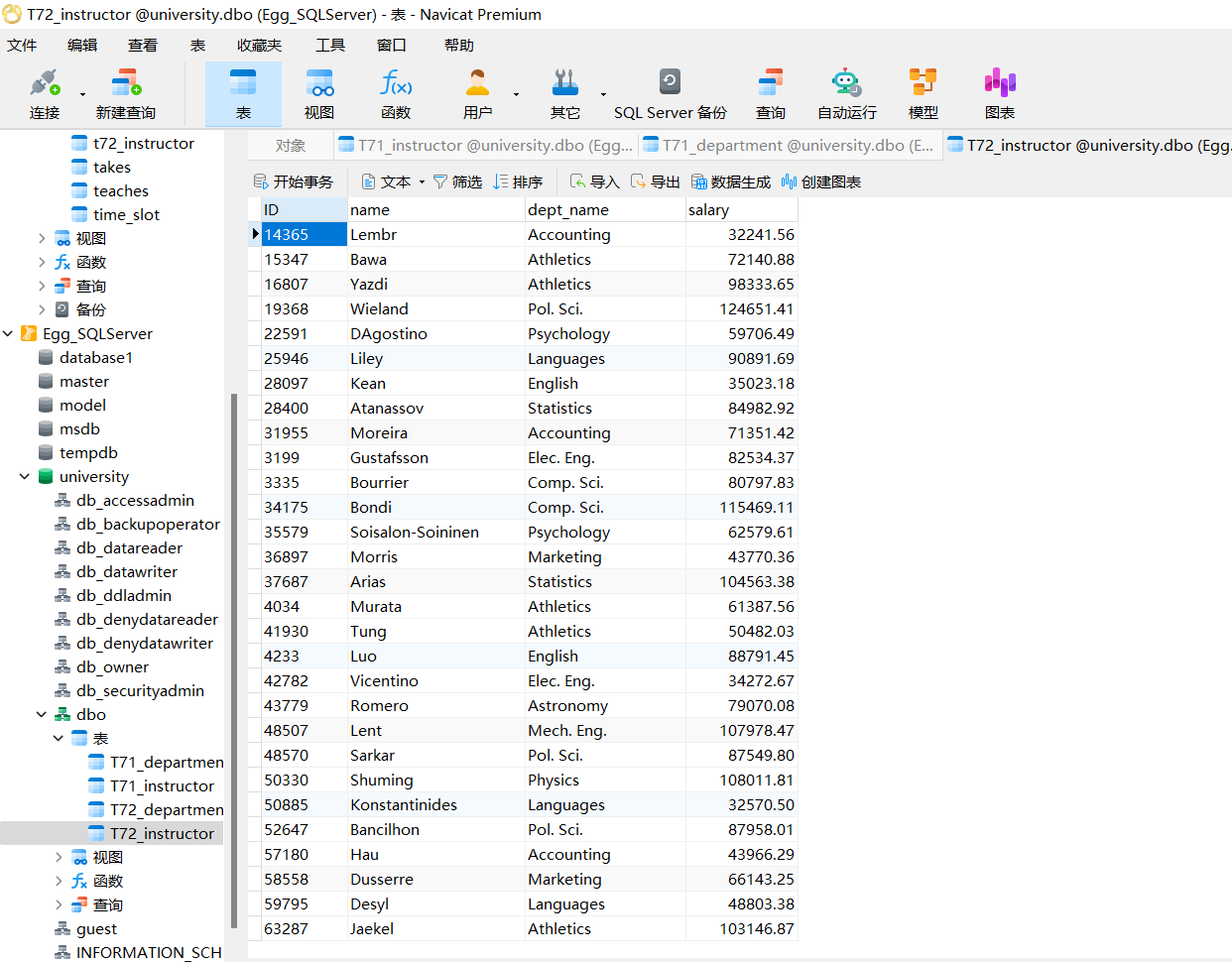
}

**Step 3：运行将显示**

验证截图1：在My SQL 数据库中的表T72\_instructor



验证截图2：在SQL Server数据库中的表T72\_instructor



* + - 1. **与实验结果相关的文件**
* 程序T72A文件
* 程序T72B文件
* 程序T72C文件
* 程序T72More1文件
  + - 1. **思考题**

(1) 尝试编写程序 T72More1 (注：参考以上程序)， 实现将数据库university中的表department从数据源university\_in\_mysql80复制到数据源university\_in\_sqlserver中的T72\_department。

package t72m;

import java.sql.\*;

public class T72More1 {

public static void main(String[] args) {

copyTableByJDBC("com.mysql.cj.jdbc.Driver"

,"jdbc:mysql://localhost:3306/university?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC"

,"root","Ling.0304",

"com.microsoft.sqlserver.jdbc.SQLServerDriver"

,"jdbc:sqlserver://localhost:1433;DatabaseName=university;"

+ "encrypt=true;"

+ "trustServerCertificate=true;"

+ "loginTimeout=30;"

,"sa","Ling.0304");

copyTableByJDBC("com.microsoft.sqlserver.jdbc.SQLServerDriver"

,"jdbc:sqlserver://localhost:1433;DatabaseName=university;"

+ "encrypt=true;"

+ "trustServerCertificate=true;"

+ "loginTimeout=30;"

,"sa","Ling.0304"

,"com.mysql.cj.jdbc.Driver"

,"jdbc:mysql://localhost:3306/university?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC"

,"root","Ling.0304"

);

}

public static void copyTableByJDBC(String src\_JDBC\_DRIVER,String src\_DB\_URL,String src\_USER,String src\_PASS,

String target\_JDBC\_DRIVER,String target\_DB\_URL,String target\_USER,String target\_PASS )

{

Connection src\_conn = null;

Statement src\_stmt = null;

Connection target\_conn = null;

PreparedStatement target\_ps=null;

try{

System.out.println("连接数据库..."+src\_DB\_URL);

System.out.println("连接数据库..."+target\_DB\_URL);

// step 1 加载驱动程序：Class.forName(driverName);

Class.forName(src\_JDBC\_DRIVER);

Class.forName(target\_JDBC\_DRIVER);

// step 2 连接数据库 ：Connection con = DriverManager.getConnection(dbURL, userName, userPwd);

src\_conn = DriverManager.getConnection(src\_DB\_URL,src\_USER,src\_PASS);

target\_conn = DriverManager.getConnection(target\_DB\_URL,target\_USER,target\_PASS);

ResultSet src\_rs;

// step 3 创建Statement/PreparedStatement对象，用来执行sql语句

// step 3.1 创建Statement： Statement stmt = con.createStatement();

// step 3.2 创建PreparedStatement对象: PreparedStatement ps=prepareStatement(sql);

src\_stmt = src\_conn.createStatement();

target\_ps = target\_conn.prepareStatement("create table T71\_department (dept\_name varchar(20) not null, building varchar(15), budget numeric(12,2) )");

target\_ps.executeUpdate();

target\_ps = target\_conn.prepareStatement("INSERT INTO T71\_department(dept\_name, building, budget) VALUES(?, ?, ?)");

//step 5 执行sql语句(接收结果集)：ResultSet rs = stmt.executeQuery(sqlStr);

String select\_sql;

select\_sql = "select dept\_name,building,budget from department";

src\_rs = src\_stmt.executeQuery(select\_sql);

// step 6 处理结果:遍历结果集

System.out.print("dept\_name\tbuilding\tbudget\n");

while(src\_rs.next()){

// 通过字段检索

String dept\_name = src\_rs.getString("dept\_name");

String building = src\_rs.getString("building");

Float budget = src\_rs.getFloat("budget");

// 输出数据

System.out.print(dept\_name);

System.out.print("\t" + building);

System.out.print("\t" + budget);

System.out.print("\n");

//step 4 给占位符赋值

target\_ps.setString(1, dept\_name); //给第1个占位符赋值(ID)

target\_ps.setString(2, building);//给第2个占位符赋值(name)

target\_ps.setFloat(3, budget);//给第3个占位符赋值(dept\_name)

//step 5.2 执行sql语句(更新操作)：ps.executeUpdate();

target\_ps.executeUpdate();

}

// step 7 完成后关闭各个对象

src\_rs.close();

src\_stmt.close();

src\_conn.close();

target\_ps.close();

target\_conn.close();

}catch(SQLException se){

// 处理 JDBC 错误

se.printStackTrace();

}catch(Exception e){

e.printStackTrace();

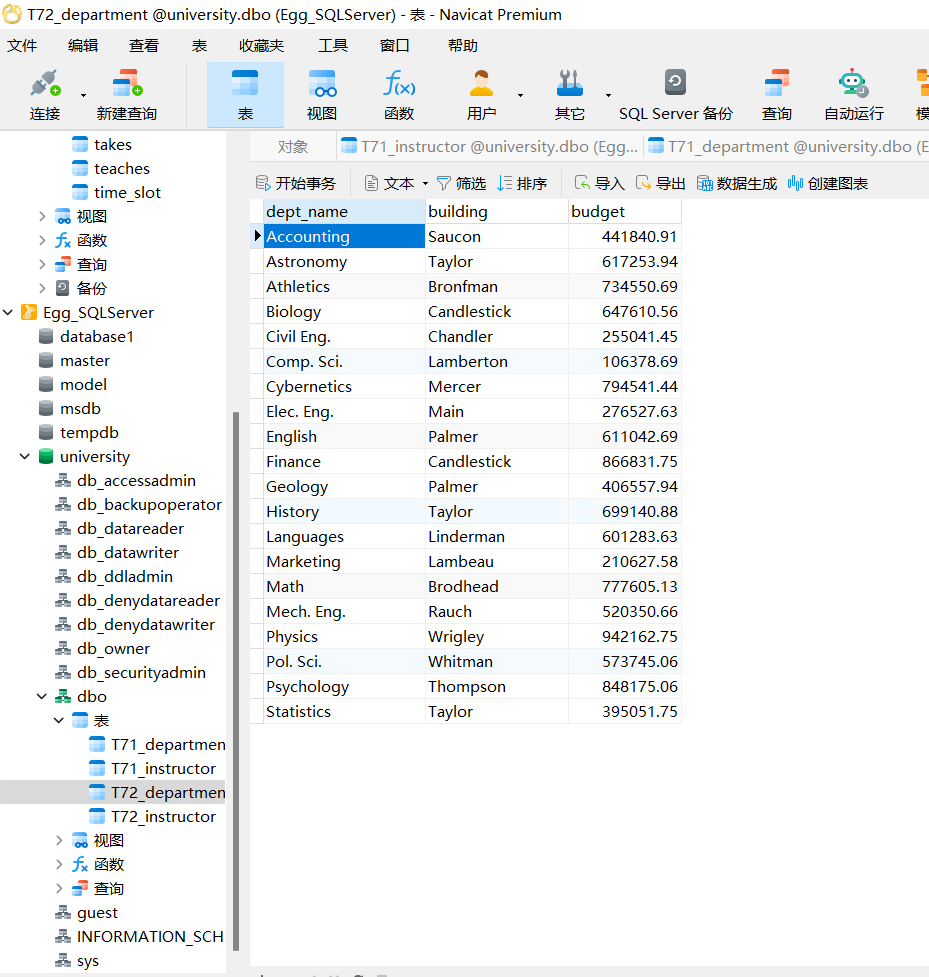
}

System.out.println("Goodbye!");

}

}

**验证截图如下**



(2) 请调查目前比较流行的软件开发环境在基于JDBC驱动开发数据库应用方面各有哪些优缺点?

答

Visual Studio Code：

优点：Visual Studio Code（VS Code）是一款轻量级且高度可配置的编辑器，支持大量扩展和插件，包括对JDBC和数据库开发的支持。它提供了良好的用户界面和便捷的代码编辑功能。

缺点：VS Code 在数据库重型操作和高级数据库调试方面可能不如专业的IDE强大。它更适合轻量级开发和快速原型设计。

JDeveloper：

优点：JDeveloper 是Oracle推出的一个集成开发环境，专注于Java应用开发，提供对JDBC的良好支持。它包含了强大的数据库工具，可以直接与Oracle数据库紧密集成。

缺点：JDeveloper 主要针对Oracle数据库优化，可能在支持其他类型数据库时不如其他IDE灵活。同时，其界面和功能对新手可能较为复杂。

DataGrip：

优点：DataGrip 是一款专门针对数据库开发和管理的IDE，提供对多种数据库的支持，包括JDBC驱动的数据库。它提供了强大的查询工具和数据操作功能。

缺点：作为一个专注于数据库的IDE，DataGrip 在通用软件开发方面的功能可能不如其他IDE全面。同时，它是一款付费软件，可能不适合预算有限的开发者。

* + - 1. **实验总结**

本次实验的目的是掌握基于JDBC驱动的数据库应用开发方法，主要涉及数据库连接和数据操作等功能。我选择了NetBeans IDE 8.2作为开发工具，配合MySQL和SQL Server数据库进行实验。

通过这次实验，我首先掌握了如何配置并使用JDBC驱动程序。下载并集成了MySQL和SQL Server的JDBC驱动程序包，这是实现数据库连接的关键步骤。我在NetBeans中创建了一个Java项目，编写了程序代码，实现了从MySQL数据库向SQL Server数据库复制数据的功能。

编写程序的过程中，我体会到了基于JDBC的数据库连接方法的重要性。实现有效的数据库连接并执行SQL语句是实验的核心。在处理SQL查询和结果集时，我学习到了如何有效地检索和处理数据。

实验中最大的挑战是熟练掌握NetBeans IDE和理解JDBC驱动的工作原理。虽然初始阶段遇到了一些困难，但通过不断尝试和调试，我逐渐克服了这些挑战。这不仅提高了我的编程技巧，也加深了我对数据库应用开发的理解。

此外，这次实验加强了我的问题解决能力，特别是在调试和优化数据库应用程序方面。通过实践，我更深刻地理解了理论知识，并将其应用于解决实际问题中。总体而言，这是一次宝贵的学习经历，为我未来的数据库项目打下了坚实的基础。