Universe Scientific Publishing
Computer System Networking and Telecommunications
计算机系统网络和电信，2019,1(2)
ISSN:2661-3719 (Print); 2661-3727 (Online)

Analysis of adding, deleting and modifying Operation of JDBC connection MySQL

Yu ZHENG Zhengde BAO Chenxi LI
School of Computer and Software, Jincheng College, Sichuan University, Chengdu,611731

Abstract
As the backbone of computer programming language, Java has an unshakable position in the field of programming. In order to store data more efficiently, the application of database such as MySQL is also necessary. A runnable network platform, not only to have a complete code framework, but also inseparable from the management of user data. How to realize the perfect connection between Java and database and how to make the best use become a hot topic for a while, and the birth of JDBC has given the answer paper for this. This paper summarizes the principle and implementation of JDBC, and analyzes how to use JDBC to connect Java program and database. And realizes the function of adding and deleting and checking.

Key Words
Java, MySQL, Database, JDBC

DOI:10.18686/jsjxt.v1i2.709

浅析 JDBC 连接 MySQL 的增删改查操作

余 峥 鲍正德 李晨曦
四川大学锦城学院计算机与软件学院，四川，成都，611731

摘 要
Java 作为计算机编程语言的中坚力量，在编程领域有着不可撼动的地位，为更高效的存储数据，像 MySQL 这样的数据库的应用也必不可少。一个可运行的网络平台，不仅要有完整的代码框架，也离不开用户数据的管理。如何让 Java 和数据库实现完美连接并发挥最佳效用成为一段时间的热议狂潮，而 JDBC 的诞生为此给出了答卷。本文概述了 JDBC 原理及实现技术方法，并浅析了如何运用 JDBC 使 Java 程序和数据库连接起来，并且实现增删查改的功能。

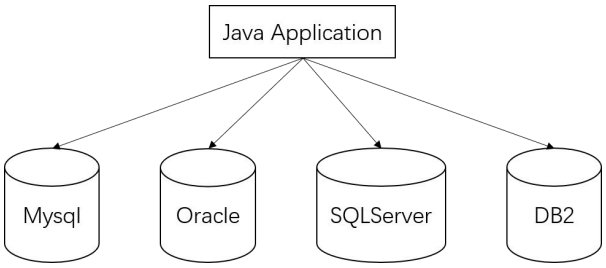
关键字
Java；MySQL；数据库；JDBC

1.对 JDBC 的理解

JDBC (Java Data Base Connectivity) 数据库访问技术是一种标准的应用程序编程接口，可以为多种关系数据库提供统一访问，[1]为了构造更高级的工具和接口，JDBC 为此提供了一种基准它能够使用用任意 Java 语言对 applet 或者 application 来进行编写, 从而在远程数据库中取得所需要的数据, 同时也可以对数据库的存取进行更新动作。JDBC 能利用简洁的操作即仅编写一次程序，然后在将 Java 和各厂商生产的多种多样的数据库相连接时仅修改少量代码, 基于这种操作，开发程序员无需对特定的数据库系统了解深入，从而简化代码的编写过程并提升了开发效率且降低了软件开发成本。

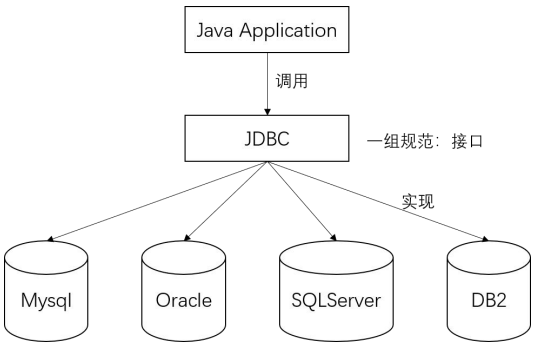
2.为什么要用到 JDBC？

在 JDBC 问世之前，每使用一种数据库，就需要用 JavaApplication 对数据库进行一次连接，而且数据库的连接方式各不相同，这就需要程序员有很高的代码水平，既费时又费力。举一个例子，如果我们用的是一个免费的 Mysql，觉得不够用了，免费升级到 Oracle，需要把底层连接数据库的方法重写，这样也不利用程序的重用。



2.1 解决方法

通过在 JAVA 应用程序和数据库之间加了一层 JDBC。JDBC 的具体实现：JAVA 直接向 JDBC 编程，JDBC 提供了一种接口和方法执行类编程，而数据库提供这种接口的实现就可以了。只要数据库想坚持 JAVA 语言的访问的话，必须提供 JDBC 这种接口的实现，这种接口的实现叫做 JDBC 的驱动。那么这种 JDBC 接口是实现是由数据库的各个厂商来完成。[2]



2.2 JDBC 驱动

数据库连接主要通过加载不同的驱动程序来完成，JDBC 驱动程序类型包括以下四种:

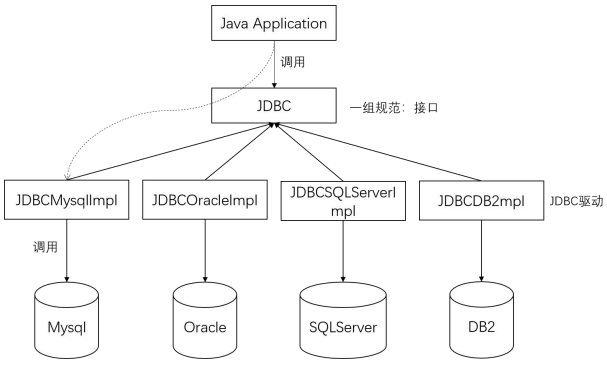
(1) JDBC-ODBC 桥加 ODBC 驱动程序:需要 ODBC 驱动。[3]

(2) 本地 API :需要驱动程序的二进制代码支持(一部分 Java 编写，一部分委托给数据库客户端代码实现)

(3) JDBC 网络纯 java 驱动程序:将 JDBC 转换为与 DBMS 无关的网络协议，又被某服务器转换为一种 DBMS 协议，以操作各种数据库 (驱动程序由中间件服务器提供)

(4) 本地协议纯 java 驱动程序:将 JDBC 调用直接转换成 JDBC 所使用的网络协议 (全部由 Java 实现，直接和数据库访访问)[3]

其中（3）和（4）所描述的是同一类的驱动程序，相比之下他们的功能、性能和可移植性都强于（1）（2）。



3.JDBC 接口、类介绍

3.1 DriverManager

JDBC 是 Java 和数据库之间连接互通的关系纽带，利用 JDBC 可使两端连接，但首先需要载入其驱动程序。

其驱动程序的命名是:com.mysql.jdbc.Driver。

具体代码如下:

Class.forName (“com.mysql.jdbc.Driver”);

该方式下和数据库连接的主要语句如下:

String jdbcUrl=” jdbc:mysql://localhost:3306/demo”;

在数据库的 JDBC URL 中 jdbc 是协议，mysql 是子协议，其后是子名称

注意：在字符串 jdbcUrl 中，demo 所代表的含义是所创建数据库的名称，这个可以自行定义。

3.2 Connection

此方法可以与数据库取得联系，造一只友谊的小船，同时为后续的操作打下基础。我们在编写了 SQL 语句之后，可以通过 Connection 去创建命令对象。创建了数据库的第一步，那么也有最后一步，就是要编写一个关闭 Connection 的方法。要记住在哪里使用了 Connection 的方法也要在那里关闭此方法。

3.3 Statement

Statement 用于来接收 Conection 类型对象方法的返回值，同时又通过此类对象调用的方法来执行 SQL 语句，在连接数据库与执行 SQL 语句之间是不可或缺必不可少的。

3.4 PreparedStatement

SQL 语句首先需要被编译，所以 DBMS 在接收到从 Prepared Statement 传来的 SQL 语句时可以直接就执行了，不用再多花时间编译。总体来说，编译 SQL 语句的时间控制在了一段时间内，空闲出来更多地时间去做其他的操作，相对的，DBMS 的工作量也减轻了许多，因此访问数据库的效率也提高了。当然有对比才会有提高，所以 Prepared Statement 对象提高数据库访问效率是建立在多次发送 SQL 语句的基础上的，如果发送的只有一次的话，那么 Statement 对象和 Prepared Statement 对象将看不出差别。[4]

3.5 ResultSet

ResultSet 简称为结果集，我们可以形象的把这个结果集想象成一个表，那么我们只想这个结果集的时候，起始位置是第一行的第一列。还可以通过循环和响应的方法去遍历结果集中的当前记录的内容。

使用 next()方法使游标可以移到下一行,并判断是否为最后一行,如果是则返回 False,否则返回 True。[5]

4 使用的软件：

本文（1）采用的数据库是 Mysql（2）开发 Java 的软件是 Eclipse（3）Mysql 数据库管理工具是 Navicat。

5.数据库

```
create database demo;
use demo;
```

5.1 创建数据库

```
create database demo;
use demo;
```

5.2 创建用户表

创建一个数据库表（table），表名为 users，其主键为 id（int）。

```
create table users(
        id int primary key auto_increment,
        name varchar(40),
        gender varchar(40)
)character set utf8 collate utf8_general_ci;
```

5.3 插入数据

向表 users 中添加 3 个数据,分别是 zhangsan、lisi、xiaohong。

```
insert into users(name,gender) values('zhangsan','boy');
insert into users(name,gender) values('lisi','boy');
insert into users(name,gender) values('xiaohong','girl');
```

6.Eclipse 相关操作

Eclipse 作为一个为 Java 定制的跨平台的自由集成开发环境，在本文中 Eclipse 的相关操作分为了六步，具体操作步骤如下：

6.1 导入 Mysql 驱动

首先在该项目下创建一个 lib 目录，把相关的 jar 包复制粘贴到 lib 目录下，右键选择 build path 选项，然后点击 add to buildpath 选项加入到类路径下。

6.2 创建 file 文件

创建一个 file 类型的配置文件，它可以用来存放少量的相关数据，可在程序代码没有加载上的时候进行应急补救。这个文件夹里面需要放入主要的文件比如 user、password 和 url，通过对配置文件中 com.mysql.jdbc.Driver 的数据库类型做一些改动来实现。通过复杂化业务的代码并多次访问，将原来在数据库中进行的逻辑计算变成业务代码中的逻辑计算，从而实现具体的数据库解耦。创建 file 文件的目的是在不修改源程序的情况下，可以获取不同数据库的连接。

driver=com.mysql.jdbc.Driver

jdbcUrl=jdbc:mysql://localhost:3306/demo

user=root

password= //密码为空

//jdbcUrl、user、password 均是准备连接数据库的基本信息

6.3 创建包 Tools

我们在 Java Resources 目录的 src 目录下创建一个包名为 Tools 的包，在该包底下创建一个类名为 JDBCTools 的工具类，其中封装一些工具方法。

6.3.1 连接数据库

通过读取配置文件从数据库服务器获取一个连接：

1）Properties：创建一个 Properties 对象，

2）InputStream：获取 file 文件的输入流。

3）Class.forName(driver):加载 mysql 驱动

4）getConnection(): 通过 DriverManager 的 getConection（）方法获取数据库的连接

```
public static Connection getConnection() throws Exception
{
        //1.  准备连接数据库的4个字符串
        String driver = null;
        String jdbcUrl = null;
        String user = null;
        String password = null;
        //1）.创建Properties 对象
        Properties properties = new Properties();
        //2） 。 获取jdbc.properties 对应的输入流
        InputStream in =
                        JDBCTools.class.getClassLoader().getResourceAsStream
        ("jdbc.properties");

        //3). 加载2) 对应的输入流
        properties.load(in);

        //4）. 具体决定user ， password 等4个字符串
        driver = properties.getProperty("driver");
        jdbcUrl = properties.getProperty("jdbcUrl");
        user = properties.getProperty("user");
        password = properties.getProperty("password");

        //2.加载数据库驱动程序（对应的Driver 实现类中有注册驱动的静态代码块）
        Class.forName(driver);

        //3. 通过DriverManager 的getConnection() 方法获取数据库连接.
        return DriverManager.getConnection(jdbcUrl, user, password);
}
```

6.3.2 关闭 Statement 和 Connection

在 JDBCTools 工具类中关闭 Statement、Connection。

```
public static void release(Statement statement,Connection conn)
        {
                if(statement!=null)
                {
                        try {
                                statement.close();
                        }catch(Exception e2){
                                e2.printStackTrace();
                        }
                }
                if(conn!=null)
                {
                        try {
                        conn.close();
                        }catch(Exception e2){
                                e2.printStackTrace();
                        }
                }
        }
```

6.3.3 添加关闭 Statement、Connection、ResultSet 对象的方法

6.3.4 通用的更新方法（增删改）

Statement 是用户执行 SQL 语句的对象

1）通过 Connection 的 createStatement（）方法来获取

2）通过 executeUpdate（sql）可以执行 SQL 语句

```
public static void release(ResultSet rs,
                Statement statement,Connection conn)
        {
                if(rs != null)
                {
                        try {
                                rs.close();
                        }catch(Exception e){
                                e.printStackTrace();
                        }
                }
                if(statement!=null)
                {
                        try {
                                statement.close();
                        }catch(Exception e2){
                                e2.printStackTrace();
                        }
                }
                if(conn!=null)
                {
                        try {
                        conn.close();
                        }catch(Exception e2){
                                e2.printStackTrace();
                        }
                }
        }
```

3）传入的 SQL 可以是 insert，update，或者 delete，但是不能为 select。

```
public void update(String sql)
        {
                Connection conn = null;
                Statement statement = null;
                try {
                        conn = JDBCTools.getConnection();
                        statement = conn.createStatement();
                        statement.executeUpdate(sql);
                }catch (Exception e)
                {
                        e.printStackTrace();
                }finally {
                        JDBCTools.release(statement, conn);
                }
        }
```

6.4 创建包 Bean

在 Java Resources 目录的 src 目录下创建一个包名为 Bean 的包，用于存放与数据库相对应的类。在创建 user 这个对象实例时，对象的属性与数据库里面的字段名对应，对象中属性的数据类型也要与数据库中的字段的类型相对应。

```
public class users {

        int id;
        String name;
        String gender;
        publicusers(int id, String name, String gender) {
        super();
        this.id = id;
        this.name = name;
        this.gender= gender;
        }
        //数据的封装
        publicintgetId() {
        return id;
        }
        publicvoidsetId(intid) {
        this.id = id;
        }
        publicStringgetName() {
        return name;
        }
        publicvoidsetName(String name) {
        this.name = name;
        }
        publicStringgetGender() {
        return gender;
        }
        publicvoidsetGender(String gender) {
        this.gender= gender;
        }
}
```

6.5 创建包 Controller

在 Java Resources 目录的 src 目录下创建一个包名为 Controller 的包，然后在该包下创建一个 JDBCTest 类，用来测试代码。在 Tools 包中有一个 update（）方法，在这个通用方法中可以进行上述三种操作。

6.5.1 添加、删除、更新操作：

添加：

```
@Test
publicvoidtestStatement() throws Exception {
String sql = "insert into users (name,gender) "+
"values('wangwu','boy');";
JDBCTools.update(sql);
}
```

删除：

```
@Test
Public void testStatement() throws Exception {
//删除
String sql= "delete from users where id = 1";
JDBCTools.update(sql);
}
```

更新：

```
@Test
publicvoidtestStatement() throws Exception {
//更新
String sql = "update users set name ='xiaoming' "+
"where id = 2";
JDBCTools.update(sql);
}
```

6.6 通过 ResultSet 执行查询操作

步骤如下：（1）首先获取 Connection（2）获取 Statement（3）准备一个 SQL 语句（4）执行查询，得到 ResultSet（5）然后处理，通过 excuteQuery 方法获取 ResultSet（6）关闭数据源。

查询操作相比于其他的三种操作与众不同的点在于执行完后会返回一个查询结果，从而达到查询信息的目的。Result Set 类的对象, 是保存 SQL 的 SE-LECT 语句返回的结果记录的表。[5]ResultSet 类的一套 get 方法(这些 get 方法可以访问当前行中的不同列) 提供了对这些行中数据的访问, 并可将结果集中的 SQL 数据类型转换为 Java 数据类型。[6]

```
@Test
public void testResultSet()
{
	Connection conn = null;
	Statement statement = null;
	ResultSet rs = null;
	try {
		//1. 获取Connection
		conn = JDBCTools.getConnection();
		//2. 获取Statement
		statement = conn.createStatement();
		//3. 准备SQL
		String sql = "SELECT * FROM users where id = 3";
		//4. 执行查询, 得到ResultSet
		rs = statement.executeQuery(sql);
		//5. 处理ResultSet
		if(rs.next())
		{
			//这里的索引是从1开始的
			int id = rs.getInt(1);
			String name = rs.getString(2);
			String gender = rs.getString(3);
			System.out.println(id);
			System.out.println(name);
			System.out.println(gender);
		}
	} catch (Exception e) {
		e.printStackTrace();
	}finally {
		//6. 关闭数据库源  这是调用JDBCTools 里面的关闭数据库的方法
		JDBCTools.release(rs, statement, conn);
	}
}
```

7.结束语

在大数据的时代背景下，互联网的应用被普及到了各个领域，所以全球数以万计的企业在开发应用的时候会使用到例如 Mysql、Oracle 等的各种数据库，JDBC 接口的出现使 Java 访问数据库变得方便了许多，JDBC 和 Java 相辅相成，使双方的作用发挥到极致。了解了 JDBC 与 Mysql 的概念后，程序员可在此基础上更深层次的探索两者之间的运行机制，然后进行一系列的 Mysql 数据库的操作。JDBC 作为优秀的访问数据库的接口，在给程序员带来便利的同时不断的降低着开发成本。

参考文献

[1]于晗.运用 JDBC 技术连接和操作 MySQL 数据库的方法[J].林区教学,2014(09):100-103.

[2]袁浩. 基于 J2EE 平台的制造业企业产品知识管理系统设计与实现[D].湖南大学,2016.

[3]张旭辉.浅析 Java 中的数据库访问[J].电子制作,2016(12):31.

[4] 戴长秀.提高 Java 访问数据库效率的方法研究与探讨[J].中国信息化,2017(06):90-92.

[5] 徐华丽.基于 JDBC 的数据库访问技术应用[J].福建电脑,2008(10):82+84.

[6]谷庆华,李成贵.基于 Java 语言实现数据库的访问[J].计算机技术与发展,2008(02):13-16.

作者简介

第一作者：余峥（1997-），男，汉，湖南省冷水江市，本科，四川大学锦城学院，研究方向：大数据技术开发，软件工程。

第二作者（通讯作者）：鲍正德（1989-），男，汉，黑龙江哈尔滨，研究生，四川大学锦城学院，研究方向：电子商务。

第三作者：李晨曦（1998-），男，汉，贵州省贵阳市，本科，四川大学锦城学院，研究方向：大数据技术开发