实验十一： 应用JDBC访问数据库

## 实验目的

1. 熟悉使用JDBC来进行数据库应用程序的设计
2. 熟悉通过JDBC接口编程访问数据库并对数据库进行操作

## 实验内容

创建一个名为Books的数据库，并在其中建立一个名为book\_message的表，字段包括书名、作者、出版社、出版时间和价格。编写一个应用程序，用预处理方式在该数据库中实现增加、删除和修改数据以及查询的基本功能。

## 实验代码

创建books表

create table Books  
*(* id varchar*(*100*)* null,  
 name varchar*(*100*)* null,  
 author varchar*(*100*)* null,  
 publisher varchar*(*100*)* null,  
 publicationDate datetime null,  
 price DOUBLE null  
*)*;

JDBCUtils.java

package com.shf.demo10;  
  
import java.lang.reflect.Field;  
import java.sql.\*;  
import java.util.ArrayList;  
import java.util.HashMap;  
import java.util.List;  
import java.util.Map;  
  
public class JDBCUtils *{  
 //数据库用户名* private static final String *USERNAME* = "root";  
 *//数据库密码* private static final String *PASSWORD* = "root";  
 *//驱动信息* private static final String *DRIVER* = "com.mysql.jdbc.Driver";  
 *//数据库地址* private static final String *URL* = "jdbc:mysql://localhost:3306/jdbc";  
 private Connection connection;  
 private PreparedStatement pstmt;  
 private ResultSet resultSet;  
  
 public JDBCUtils*() {* try*{* Class.*forName(DRIVER)*;  
 System.*out*.println*(*"数据库连接成功！"*)*;  
 connection = getConnection*()*;  
 *}*catch*(*Exception e*){* e.printStackTrace*()*;  
 *}  
 }  
  
 /\*\*  
 \* 获得数据库的连接  
 \* @return  
 \*/* public Connection getConnection*(){* try *{* connection = DriverManager.*getConnection(URL*, *USERNAME*, *PASSWORD)*;  
 *}* catch *(*SQLException e*) {* e.printStackTrace*()*;  
 *}* return connection;  
 *}  
  
 /\*\*  
 \* 释放数据库连接  
 \*/* public void releaseConn*(){* if*(*resultSet != null*){* try*{* resultSet.close*()*;  
 *}*catch*(*SQLException e*){* e.printStackTrace*()*;  
 *}  
 }  
 }  
  
 /\*\*  
 \* 增加、删除、改  
 \* @param sql  
 \* @param params  
 \* @return  
 \* @throws SQLException  
 \*/* public boolean updateByPreparedStatement*(*String sql, List*<*Object*>* params*)*throws SQLException*{* boolean flag = false;  
 int result = -1;  
 pstmt = connection.prepareStatement*(*sql*)*;  
 int index = 1;  
 if*(*params != null && !params.isEmpty*()){* for*(*int i=0; i<params.size*()*; i++*){* pstmt.setObject*(*index++, params.get*(*i*))*;  
 *}  
 }* result = pstmt.executeUpdate*()*;  
 flag = result > 0 ? true : false;  
 releaseConn*()*;  
 return flag;  
 *}  
  
 /\*\*  
 \* 查询单条记录  
 \* @param sql  
 \* @param params  
 \* @return  
 \* @throws SQLException  
 \*/* public Map*<*String, Object*>* findSimpleResult*(*String sql, List*<*Object*>* params*)* throws SQLException*{* Map*<*String, Object*>* map = new HashMap*<*String, Object*>()*;  
 int index = 1;  
 pstmt = connection.prepareStatement*(*sql*)*;  
 if*(*params != null && !params.isEmpty*()){* for*(*int i=0; i<params.size*()*; i++*){* pstmt.setObject*(*index++, params.get*(*i*))*;  
 *}  
 }* resultSet = pstmt.executeQuery*()*;*//返回查询结果* ResultSetMetaData metaData = resultSet.getMetaData*()*;  
 int col\_len = metaData.getColumnCount*()*;  
 while*(*resultSet.next*()){* for*(*int i=0; i<col\_len; i++ *){* String cols\_name = metaData.getColumnName*(*i+1*)*;  
 Object cols\_value = resultSet.getObject*(*cols\_name*)*;  
 if*(*cols\_value == null*){* cols\_value = "";  
 *}* map.put*(*cols\_name, cols\_value*)*;  
 *}  
 }* releaseConn*()*;  
 return map;  
 *}  
  
 /\*\*查询多条记录  
 \* @param sql  
 \* @param params  
 \* @return  
 \* @throws SQLException  
 \*/* public List*<*Map*<*String, Object*>>* findModeResult*(*String sql, List*<*Object*>* params*)* throws SQLException*{* List*<*Map*<*String, Object*>>* list = new ArrayList*<*Map*<*String, Object*>>()*;  
 int index = 1;  
 pstmt = connection.prepareStatement*(*sql*)*;  
 if*(*params != null && !params.isEmpty*()){* for*(*int i = 0; i<params.size*()*; i++*){* pstmt.setObject*(*index++, params.get*(*i*))*;  
 *}  
 }* resultSet = pstmt.executeQuery*()*;  
 ResultSetMetaData metaData = resultSet.getMetaData*()*;  
 int cols\_len = metaData.getColumnCount*()*;  
 while*(*resultSet.next*()){* Map*<*String, Object*>* map = new HashMap*<*String, Object*>()*;  
 for*(*int i=0; i<cols\_len; i++*){* String cols\_name = metaData.getColumnName*(*i+1*)*;  
 Object cols\_value = resultSet.getObject*(*cols\_name*)*;  
 if*(*cols\_value == null*){* cols\_value = "";  
 *}* map.put*(*cols\_name, cols\_value*)*;  
 *}* list.add*(*map*)*;  
 *}* releaseConn*()*;  
 return list;  
 *}  
}*

Book.java

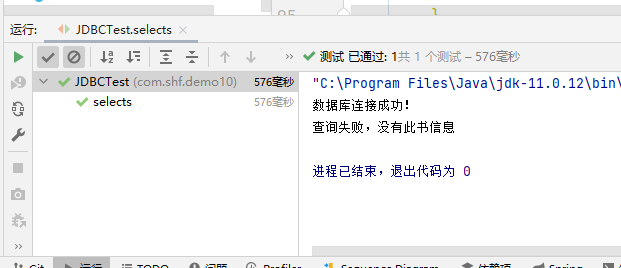
package com.shf.demo10;  
  
import java.util.Date;  
  
public class Book {  
 private String id;  
 private String name;  
 private String author;  
 private String publisher;  
 private Date publicationDate;  
 private Double price;  
  
 public Book() {  
 }  
  
 public Book(String id, String name, String author, String publisher, Date publicationDate, Double price) {  
 this.id = id;  
 this.name = name;  
 this.author = author;  
 this.publisher = publisher;  
 this.publicationDate = publicationDate;  
 this.price = price;  
 }  
  
 public String getId() {  
 return id;  
 }  
  
 public void setId(String id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getAuthor() {  
 return author;  
 }  
  
 public void setAuthor(String author) {  
 this.author = author;  
 }  
  
 public String getPublisher() {  
 return publisher;  
 }  
  
 public void setPublisher(String publisher) {  
 this.publisher = publisher;  
 }  
  
 public Date getPublicationDate() {  
 return publicationDate;  
 }  
  
 public void setPublicationDate(Date publicationDate) {  
 this.publicationDate = publicationDate;  
 }  
  
 public Double getPrice() {  
 return price;  
 }  
  
 public void setPrice(Double price) {  
 this.price = price;  
 }  
  
 @Override  
 public String toString() {  
 return "Book{" +  
 "id=" + id +  
 ", name='" + name + '\'' +  
 ", author='" + author + '\'' +  
 ", publisher='" + publisher + '\'' +  
 ", publicationDate=" + publicationDate +  
 ", price=" + price +  
 '}';  
 }  
}

JDBCTest.java

package com.shf.demo10;  
  
import org.junit.Test;  
  
import java.sql.SQLException;  
import java.util.\*;  
  
public class JDBCTest *{* JDBCUtils jdbcUtils = new JDBCUtils*()*;  
  
 @Test  
 public void insert*()* throws SQLException *{* String sql = "insert into books (id,name, author,publisher,publicationDate,price) values (?,?,?,?,?,?), (?,?,?,?,?,?), (?,?,?,?,?,?)";  
 List*<*Object*>* params = new ArrayList*<>()*;  
 Book b1 = new Book*(*"9787212058937", "有理想就有疼痛", "高晓春", "安徽人民出版社", new Date*(*2019 / 12 / 20*)*, 29.00*)*;  
 Book b2 = new Book*(*"9787115402547","SVG精髓","(美) 艾森伯格 (Eisenberg,J.D.)","人民邮电出版社",new Date*(*2020/12/20*)*,71.24*)*;  
 Book b3 = new Book*(*"9787115429674","Linux命令行与shell脚本编程大全","(美) 布鲁姆 (Richard Blum) , (美)","人民邮电出版社",new Date*(*2209/12/20*)*,99.19*)*;  
 params.add*(*b1.getId*())*;  
 params.add*(*b1.getName*())*;  
 params.add*(*b1.getAuthor*())*;  
 params.add*(*b1.getPublisher*())*;  
 params.add*(*b1.getPublicationDate*())*;  
 params.add*(*b1.getPrice*())*;  
 params.add*(*b2.getId*())*;  
 params.add*(*b2.getName*())*;  
 params.add*(*b2.getAuthor*())*;  
 params.add*(*b2.getPublisher*())*;  
 params.add*(*b2.getPublicationDate*())*;  
 params.add*(*b2.getPrice*())*;  
 params.add*(*b3.getId*())*;  
 params.add*(*b3.getName*())*;  
 params.add*(*b3.getAuthor*())*;  
 params.add*(*b3.getPublisher*())*;  
 params.add*(*b3.getPublicationDate*())*;  
 params.add*(*b3.getPrice*())*;  
 boolean flag = jdbcUtils.updateByPreparedStatement*(*sql,params*)*;  
 System.*out*.println*(*flag*)*;  
 if *(*flag*){* System.*out*.println*(*"添加成功，添加的书籍信息为："+b1+""+b2+""+b3*)*;  
 *}* else *{* System.*out*.println*(*"添加失败"*)*;  
 *}  
 }* @Test  
 public void delete*()* throws SQLException *{* String sql = "delete from books where id = ?";  
 List*<*Object*>* params = new ArrayList*<*Object*>()*;  
 params.add*(*"9787212058937"*)*;  
 boolean flag = jdbcUtils.updateByPreparedStatement*(*sql, params*)*;  
 System.*out*.println*(*flag*)*;  
 if *(*flag*){* System.*out*.println*(*"删除成功，删除的书籍ID为："+params.get*(*0*))*;  
 *}* else *{* System.*out*.println*(*"删除失败"*)*;  
 *}  
 }* @Test  
 public void alter*()* throws SQLException *{* String sql = "update books set price = ? where id = ? ";  
 List*<*Object*>* params = new ArrayList*<*Object*>()*;  
 params.add*(*"99.9"*)*;  
 params.add*(*"9787212058937"*)*;  
 boolean flag = jdbcUtils.updateByPreparedStatement*(*sql, params*)*;  
 System.*out*.println*(*flag*)*;  
 if *(*flag*){* System.*out*.println*(*"修改成功，修改的书籍ID为："+params.get*(*1*))*;  
 *}* else *{* System.*out*.println*(*"修改失败"*)*;  
 *}  
 }* @Test  
 public void selects*()* throws SQLException *{* String sql = "select \* from books";  
 List*<*Map*<*String, Object*>>* list = jdbcUtils.findModeResult*(*sql, null*)*;  
 if *(*list.size*()*!=0*){* System.*out*.println*(*"查询成功，书籍的信息为："+list*)*;  
 *}* else *{* System.*out*.println*(*"查询失败，没有此书信息"*)*;  
 *}  
 }* @Test  
 public void select*()* throws SQLException *{* String sql = "select \* from books where id=?";  
 List*<*Object*>* params = new ArrayList*<*Object*>()*;  
 params.add*(*"9787115429674"*)*;  
 Map*<*String, Object*>* list = jdbcUtils.findSimpleResult*(*sql, params*)*;  
 if *(*list.size*()*!=0*){* System.*out*.println*(*"查询成功，书籍的信息为："+list*)*;  
 *}* else *{* System.*out*.println*(*"查询失败，没有此书信息"*)*;  
 *}  
 }  
}*

## 实验截图

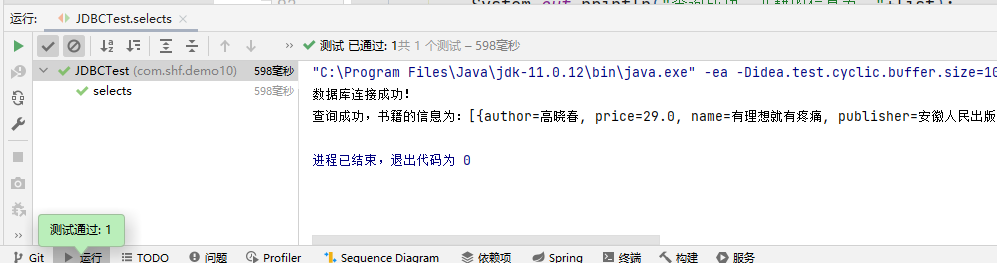
查询所有书籍

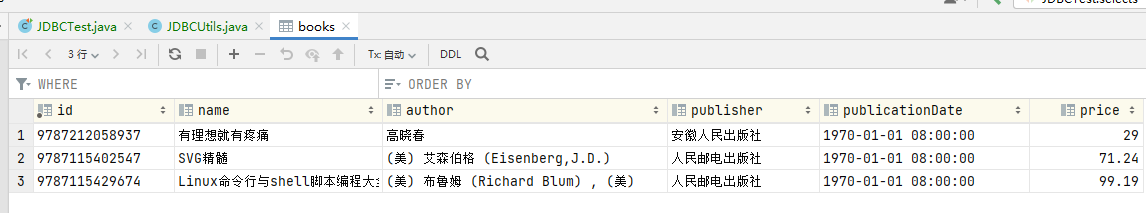


批量添加书籍

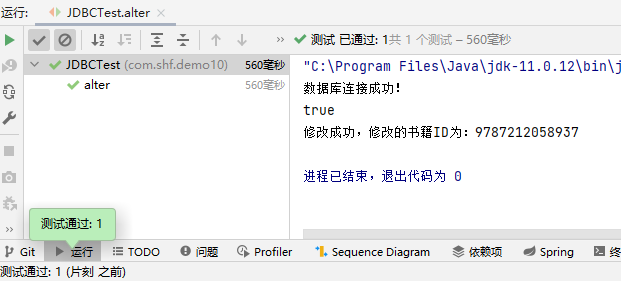


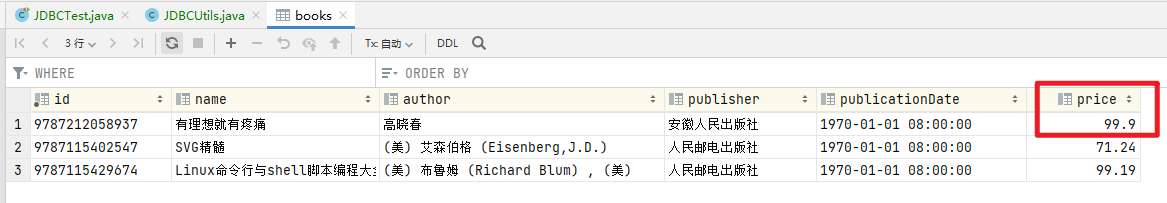
查询所有插入的书籍



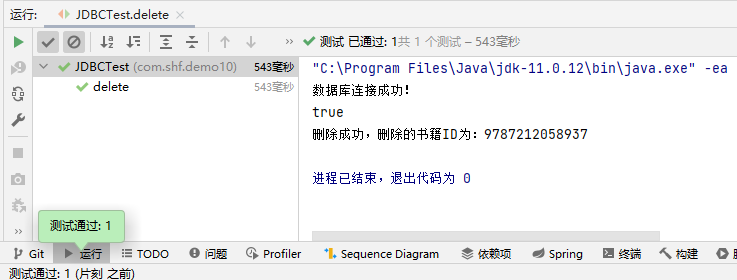


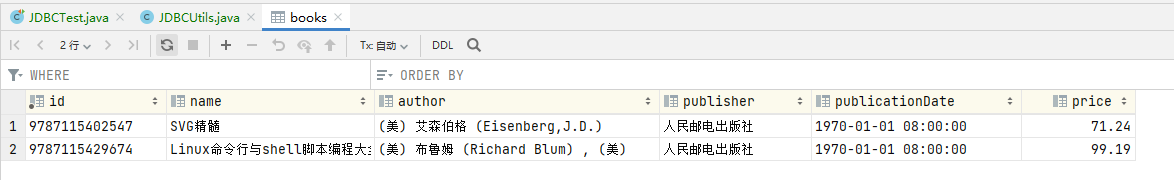
通过id修改书籍价格



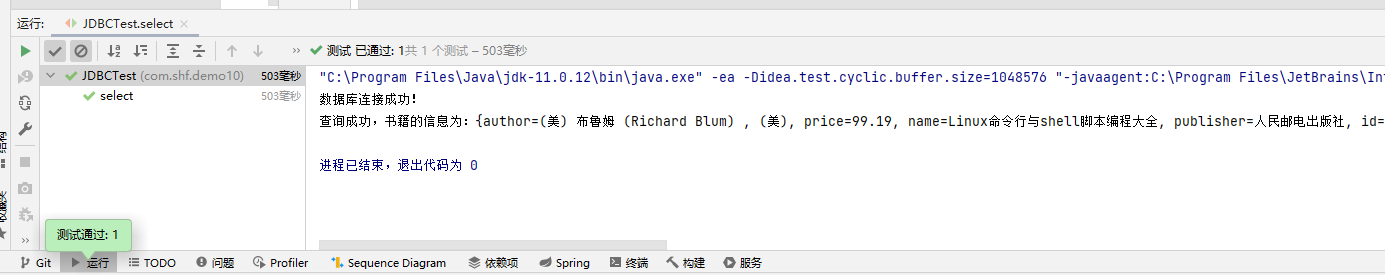


通过id删除书籍





查询一条数据



## 实验小结

通过本次实验我学会使用JDBC + JavaBean + MySQL连接数据库，给数据库进行基本的增删改查应用，使用junit对返回进行测试，并且封装成为JDBCUtils工具类，往工具类中传入sql语句和args参数就可以执行增删改查。