实验七 JDBC进阶

一、相关知识点

1. JDBC基本概念
2. 主从关系，分页查询，连接池

二、实验目的

理解分页查询的概念和处理方法

三、实验内容

1、 数据准备：编写程序，自动生成1000个读者和图书；

```
//num为需要创建读者的数量，PreReaderTypeId为读者类型ID
public void AutogenerationReaders(int num, int PreReaderTypeId) throws BaseException{
        Connection conn = null;
        java.sql.PreparedStatement pst = null;
        try {
            conn = DBUtil.getConnection();
            String sql = "select lendBookLimitted from BeanReaderType where readerTypeId=" + PreReaderTypeId;
            java.sql.Statement st = conn.createStatement();
            java.sql.ResultSet rs = st.executeQuery(sql);
            if (!rs.next()) throw new BusinessException("读者类别不存在");
            int lendBookLimitted = rs.getInt(1);
            rs.close();
            st.close();
```

```
14          sql = "insert into BeanReader(readerid,readerName,readerTypeId,lendBookLimitted,createDate,creatorUserId) values(?,?,?,?,?,?)";
15          pst = conn.prepareStatement(sql);
16          //手动事务
17          conn.setAutoCommit(false);
18          Long startTime = System.currentTimeMillis();
19          int count =0;
20          System.out.println("开始插入...");
21          for(int i = 0; i < num; i++){
22              pst.setString(1, String.valueOf(i));
23              pst.setString(2, String.valueOf(i));
24              pst.setInt(3, PreReaderTypeId);//将所有创建的读者类型设为
25              pst.setInt(4, lendBookLimitted);
26              pst.setTimestamp(5, new java.sql.Timestamp(System.currentTimeMillis()));
27              pst.setString(6, "admin"); //默认为admin创建
28              pst.addBatch();
29              count++;
30              if(count>=25000) {
31                  //每25000条数据进行一次批量插入操作
32                  pst.executeBatch();
33                  pst.clearBatch();
34                  conn.commit();
35                  count = 0;
36              }
37          }
38          if(count != 0)
39          {
40              pst.executeBatch();
41              pst.clearBatch();
42              conn.commit();
43              count = 0;
44          }
45          Long endTime = System.currentTimeMillis();
46          System.out.println(num + "条数据插入完成,总用时：" + (endTime - startTime)+"ms");
47      } catch (Exception e) {
48          e.printStackTrace();
49          throw new RuntimeException(e);
50      }finally{
51          if(pst !=null){
```

```
            try {
                pst.close();
            } catch (SQLException e) {
                e.printStackTrace();
                throw new RuntimeException(e);
            }
        }
        if(conn!=null){
            try {
                conn.close();
            } catch (SQLException e) {
                e.printStackTrace();
                throw new RuntimeException(e);
            }
        }
    }
}
```

```
//num为需要创建书本的数量，Prepubid为出版社ID
public void AutogenerationBooks(int num, String Prepubid) throws BaseException{
    Connection conn = null;
    java.sql.PreparedStatement pst = null;
    try {
        conn = DBUtil.getConnection();
        String sql = "select * from beanpublisher where pubid=" + Prepubid;
        java.sql.Statement st = conn.createStatement();
        java.sql.ResultSet rs = st.executeQuery(sql);
        if (!rs.next()) throw new BusinessException("出版社类别不存在");
        rs.close();
        st.close();
        sql = "insert into BeanBook(barcode,bookname,pubid,price,state,storagetime) values(?,?,?,?,'在库',?)";
        pst = conn.prepareStatement(sql);
        //手动事务
        conn.setAutoCommit(false);
        Long startTime = System.currentTimeMillis();
        int count =0;
        System.out.println("开始插入...");
        for(int i = 0; i < num; i++){
            pst = conn.prepareStatement(sql);
            pst.setString(1, String.valueOf(i));
```

```
23                pst.setString(2, String.valueOf(i));
24                pst.setString(3, Prepubid);
25                pst.setDouble(4, 10);
26                pst.setTimestamp(5, new
java.sql.Timestamp(System.currentTimeMillis()));
27                pst.addBatch();
28                count++;
29                if(count>=25000) {
30                    //每25000条数据进行一次批量插入操作
31                    pst.executeBatch();
32                    pst.clearBatch();
33                    conn.commit();
34                    count = 0;
35                }
36            }
37            if(count != 0)
38            {
39                pst.executeBatch();
40                pst.clearBatch();
41                conn.commit();
42                count = 0;
43            }
44            Long endTime = System.currentTimeMillis();
45            System.out.println(num + "条数据插入完成,总用时：" + (endTime -
startTime)+"ms");
46        } catch (Exception e) {
47            e.printStackTrace();
48            throw new RuntimeException(e);
49        }finally{
50            if(pst !=null){
51                try {
52                    pst.close();
53                } catch (SQLException e) {
54                    e.printStackTrace();
55                    throw new RuntimeException(e);
56                }
57            }
58            if(conn!=null){
59                try {
60                    conn.close();
61                } catch (SQLException e) {
62                    e.printStackTrace();
```

```
                throw new RuntimeException(e);
            }
        }
    }
}
```

```
mysql> select * from beanbook;
+-----------+----------+-------+-------+--------+---------------------+
| barcode   | bookname | pubid | price | state  | storagetime         |
+-----------+----------+-------+-------+--------+---------------------+
| 0         | 0        | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 1         | 1        | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 10        | 10       | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 100       | 100      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 101       | 101      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 102       | 102      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 103       | 103      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 104       | 104      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 105       | 105      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 106       | 106      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 107       | 107      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 108       | 108      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 109       | 109      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 11        | 11       | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 110       | 110      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 111       | 111      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 112       | 112      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 113       | 113      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 114       | 114      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 115       | 115      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 116       | 116      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 117       | 117      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 118       | 118      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 119       | 119      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 12        | 12       | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
| 120       | 120      | 1     |    10 | 在库   | 2022-05-22 19:53:44 |
```

```
mysql> select * from beanreader;
+---------+----------+-------------+-----------------+---------------------+-------------+-----------+-------------+---------+-----------+
| readerid | readerName | readerTypeId | lendBookLimitted | createDate          | creatorUserId | removeDate | removerUserId | stopDate | stopUserId |
+---------+----------+-------------+-----------------+---------------------+-------------+-----------+-------------+---------+-----------+
| 0       | 0        |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 1       | 1        |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 10      | 10       |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 100     | 100      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 101     | 101      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 102     | 102      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 103     | 103      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 104     | 104      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 105     | 105      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 106     | 106      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 107     | 107      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 108     | 108      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 109     | 109      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 11      | 11       |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 110     | 110      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 111     | 111      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 112     | 112      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 113     | 113      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 114     | 114      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 115     | 115      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 116     | 116      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 117     | 117      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 118     | 118      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 119     | 119      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 12      | 12       |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
| 120     | 120      |           8 |               6 | 2022-05-22 16:23:08 | admin       | NULL      | NULL        | NULL    | NULL      |
```

2、 改造读者模块，在提取读者的同时，提取其未归还的图书信息

第一步：通过程序增加一些借阅纪录

第二步：改造读者javabean，使之包括借阅的图书信息

第三步：改造ReaderManager中的读者提取方法（public BeanReader loadReader(String readerid) throws DbException ），同时提取未归还图书；

第四步：ReaderManager的main函数中调用该方法进行测试，要求输出指定读者的基本信息及其未归还的图书名称。

【实验结果与分析】

A、 javabean类代码。

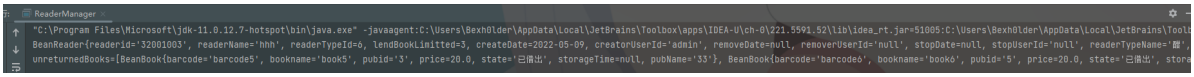
```
private List<BeanBook> unreturnedBooks = new ArrayList<BeanBook>();

public List<BeanBook> getUnreturnedBooks() {
        return unreturnedBooks;
}

public void setUnreturnedBooks(List<BeanBook> unreturnedBooks) {
    this.unreturnedBooks = unreturnedBooks;
}
```

B、 给出改造后ReaderManager类的方法代码。

```
public BeanReader loadReader(String readerid) throws DbException {
        Connection conn = null;
        try {
            conn = DBUtil.getConnection();
            String sql = "select readerid,readerName,r.readerTypeId,r.lendBookLimitted,createDate,creatorUserId,stopDate,stopUserId,rt.readerTypeName,r.removeDate" +
                    "  from BeanReader r,BeanReaderType rt where r.readerTypeId=rt.readerTypeId" +
                    " and  r.readerid=?";
            sql += " order by readerid";
            java.sql.PreparedStatement pst = conn.prepareStatement(sql);
            pst.setString(1, readerid);
            java.sql.ResultSet rs = pst.executeQuery();
            if (rs.next()) {
                BeanReader r = new BeanReader();
                r.setReaderid(rs.getString(1));
                r.setReaderName(rs.getString(2));
                r.setReaderTypeId(rs.getInt(3));
                r.setLendBookLimitted(rs.getInt(4));
                r.setCreateDate(rs.getDate(5));
                r.setCreatorUserId(rs.getString(6));
                r.setStopDate(rs.getDate(7));
                r.setStopUserId(rs.getString(8));
                r.setReaderTypeName(rs.getString(9));
                r.setRemoveDate(rs.getDate(10));
```

```java
            r.setUnreturnedBooks((new BookLendManager()).loadReaderLentBooks(r.getReaderid()));//修改处
            return r;
        }
    } catch (SQLException e) {
        e.printStackTrace();
        throw new DbException(e);
    } finally {
        if (conn != null)
            try {
                conn.close();
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
    }
    return null;
}
```

```java
public static void main(String[] args) {
    try {
        BeanReader test = (new ReaderManager()).loadReader("32001003");
        System.out.println(test);
    } catch (BaseException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
//  try {
//      pm.deletePublisher("testpubid");
//  } catch (BaseException e) {
//      // TODO Auto-generated catch block
//      e.printStackTrace();
//  }
}
```



3、 改造读者管理模块，将读者列表页面改造成分页查询方式。

第一步：自行设计PageData类，用于存放分页数据

第二步：改造ReaderManager类，将其中的查询读者方法改造成分页查询。

第三步（选做）：修改ui类，增加上一页、下一页按钮，实现读者的分页查询，要求每页20人

【实验结果与分析】

A、 PageData类代码。

```java
package cn.edu.zucc.booklib.model;

import java.util.ArrayList;

public class PageData {
    private int totalRecordCount;
    private static int pageSize = 10;
    private static int pageIndex = 1;
    private ArrayList<BeanReader> beanReaders = new ArrayList<>(pageSize);

    public PageData(int pageSize) {
        this.pageSize = pageSize;
    }

    public int getTotalRecordCount() {
        return totalRecordCount;
    }

    public void setTotalRecordCount(int totalRecordCount) {
        this.totalRecordCount = totalRecordCount;
    }

    public static int getPageSize() {
        return pageSize;
    }

    public static void setPageSize(int pageSize) {
        PageData.pageSize = pageSize;
    }

    public static int getPageIndex() {
        return pageIndex;
    }

    public static void setPageIndex(int pageIndex) {
        PageData.pageIndex = pageIndex;
```

```
    }

    public PageData() {
    }

    public ArrayList<BeanReader> getBeanReaders() {
        return beanReaders;
    }

    public void setBeanReaders(ArrayList<BeanReader> beanReaders) {
        this.beanReaders = beanReaders;
    }
}
```

B、 给出改造后ReaderManager类的方法代码。

```
public List<BeanReader> searchReader(String keyword, int readerTypeId) throws BaseException {
        List<BeanReader> result = new ArrayList<BeanReader>();
        Connection conn = null;
        try {
            conn = DBUtil.getConnection();
            String sql = "select readerid,readerName,r.readerTypeId,r.lendBookLimitted,createDate,creatorUserId,stopDate,stopUserId,rt.readerTypeName" +
                    "  from BeanReader r,BeanReaderType rt where r.readerTypeId=rt.readerTypeId" +
                    " and removeDate is null ";
            if (readerTypeId > 0) sql += " and r.readerTypeId=" + readerTypeId;
            if (keyword != null && !"".equals(keyword))
                sql += " and (readerid like ? or readerName like ?)";
            sql += " order by readerid";
            //-----------------------------------------------------------------------
            sql += "limit ?,?";
            java.sql.PreparedStatement pst = conn.prepareStatement(sql);
            if (keyword != null && !"".equals(keyword)) {
                pst.setString(1, "%" + keyword + "%");
                pst.setString(2, "%" + keyword + "%");
                pst.setObject(3, (PageData.getPageIndex()-1)*PageData.getPageSize());
                pst.setObject(4,PageData.getPageSize());
            }else {
```

```
            pst.setObject(1, (PageData.getPageIndex()-1)*PageData.getPageSize());
            pst.setObject(2,PageData.getPageSize());
        }
    //-----------------------------------------------------------------------
        java.sql.ResultSet rs = pst.executeQuery();
        while (rs.next()) {
            BeanReader r = new BeanReader();
            r.setReaderid(rs.getString(1));
            r.setReaderName(rs.getString(2));
            r.setReaderTypeId(rs.getInt(3));
            r.setLendBookLimitted(rs.getInt(4));
            r.setCreateDate(rs.getDate(5));
            r.setCreatorUserId(rs.getString(6));
            r.setStopDate(rs.getDate(7));
            r.setStopUserId(rs.getString(8));
            r.setReaderTypeName(rs.getString(9));
            result.add(r);
        }
    } catch (SQLException e) {
        e.printStackTrace();
        throw new DbException(e);
    } finally {
        if (conn != null)
            try {
                conn.close();
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
    }
    return result;

}
```

C、 给出ui类中的修改部分（注：生成表格的方法需做微调）

4、 用C3P0连接池改造booklib应用

第一步：将mchange-commons-java-0.2.3.4、c3p0-0.9.2.1.jar引入工程

第二步：在cn.edu.zucc.booklib.util包下增加类DBUtil2，并模仿DBPool类实现DBUtil类中定义的功能

第三步：改造BookManager类，将其各方法中获取数据库的连接的方法改成用DBUtil2；

第四步：BookManager中编写main函数，利用循环待用添加图书方法的形式增加1000本图书，并记录整体运行时间；分别测试利用DBUtil和DBUtil2获取数据库连接的耗时。

【实验结果与分析】

A、 DBUtil2类代码。

```
package cn.edu.zucc.booklib.util;

import com.mchange.v2.c3p0.ComboPooledDataSource;

import java.beans.PropertyVetoException;
import java.sql.*;

public class DBUtil2 {
    private static final String jdbcDriver = "com.mysql.jdbc.Driver";
    private static final String jdbcUrl = "jdbc:mysql://localhost:3306/booklib?useUnicode=true&characterEncoding=UTF-8";
    private static final String dbUser = "root";
    private static final String dbPwd = "xxbbhh";


    public static Connection getConnection() throws java.sql.SQLException {
        ComboPooledDataSource cpds = new ComboPooledDataSource();
        try {
            cpds.setDriverClass(jdbcDriver);
            cpds.setJdbcUrl(jdbcUrl);
            cpds.setUser(dbUser);
            cpds.setPassword(dbPwd);
            return cpds.getConnection();
        } catch (Exception e) {
            throw new ExceptionInInitializerError(e);
        }

    }

    public static void closeResorce(Connection conn, Statement ps) {
```

```
        try {
            if (conn != null)
                conn.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        try {
            if (ps != null)
                ps.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public static void closeResorce(Connection conn, Statement ps, ResultSet rs) {
        try {
            if (conn != null)
                conn.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        try {
            if (ps != null)
                ps.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }

        try {
            if (rs != null)
                rs.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

B、 给出改造后BookManager类的main函数代码。

```
public void insertBooks_DBUtil() {
        Connection conn = null;
        java.sql.PreparedStatement ps = null;
```

```
4        try {
5            conn = DBUtil.getConnection();
6
7            String sql = "insert into BeanBook(barcode,bookname,pubid,price,state)
values(?,?,?,?,'在库')";
8            ps = conn.prepareStatement(sql);
9
10           for (int i = 2000; i < 3000; i++) {
11               ps.setObject(1, i);
12               ps.setObject(2, "book_" + i);
13               ps.setObject(3, 5);
14               ps.setObject(4, 10);
15               ps.execute();
16           }
17       } catch (SQLException e) {
18           e.printStackTrace();
19       } finally {
20           DBUtil.closeResorce(conn, ps);
21       }
22
23   }
24   public void insertBooks_DBUtil2() {
25       Connection conn = null;
26       java.sql.PreparedStatement ps = null;
27       try {
28           conn = DBUtil2.getConnection();
29
30           String sql = "insert into BeanBook(barcode,bookname,pubid,price,state)
values(?,?,?,?,'在库')";
31           ps = conn.prepareStatement(sql);
32
33           for (int i = 1000; i < 2000; i++) {
34               ps.setObject(1, i);
35               ps.setObject(2, "book_" + i);
36               ps.setObject(3, 5);
37               ps.setObject(4, 10);
38               ps.execute();
39           }
40       } catch (SQLException e) {
41           e.printStackTrace();
42       } finally {
43           DBUtil.closeResorce(conn, ps);
```

```java
        }

    }
    public static void main(String[] args) {
        try {
            long startTime = System.currentTimeMillis();
            (new BookManager()).insertBooks_DBUtil();
            long endTime = System.currentTimeMillis();
            System.out.println("DBUtil使用时间：  " + (endTime - startTime));
            startTime = System.currentTimeMillis();
            (new BookManager()).insertBooks_DBUtil2();
            endTime = System.currentTimeMillis();
            System.out.println("DBUtil2使用时间：  " + (endTime - startTime));
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
```

C、 给出用DBUtil和DBUtil2获取数据库连接时，main函数的执