实验八 JDBC进阶（2）

一、相关知识点

1. JDBC基本概念
2. 视图、索引
3. java集合框架

二、实验目的：

1. 理解视图和索引的概念，并通过视图设计简化程序设计，通过索引设计优化查询性能；
2. 理解java集合框架

三、实验内容：

1、 设计读者视图view_reader，并修改readerManager类中相关代码。

第一步：建立读者视图，要求视图中包含读者类别名称；

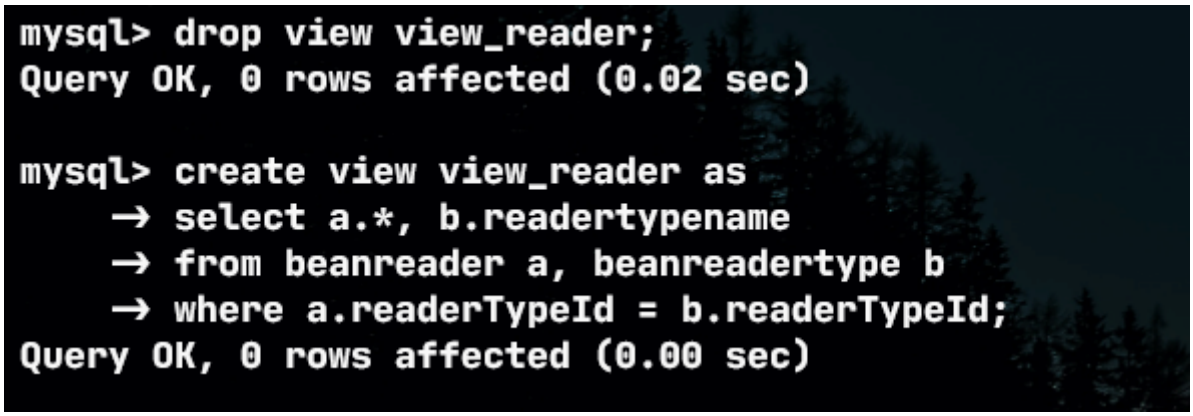
第二步：改造ReaderManager类，将其中的连接查询用视图代替。

第三步：运行图书管理系统，进行各个功能的测试（读者类别管理、读者管理）

【实验结果与分析】

A、 写出视图创建代码。

```
create view view_reader as
select a.*, b.readertypename
from beanreader a, beanreadertype b
where a.readerTypeId = b.readerTypeId;
```



B、 给出改造后ReaderManager类的各个方法的代码。

- ReaderManager.searchReader

```
1  public List<BeanReader> searchReader(String keyword, int readerTypeId)
   throws BaseException {
2      List<BeanReader> result = new ArrayList<BeanReader>();
3      Connection conn = null;
4      try {
5          conn = DBUtil.getConnection();
6          //连接查询
7  //          String sql = "select
   readerid,readerName,r.readerTypeId,r.lendBookLimitted,createDate,creatorUserI
   d,stopDate,stopUserId,rt.readerTypeName" +
8  //                  "  from BeanReader r,BeanReaderType rt where
   r.readerTypeId=rt.readerTypeId" +
9  //                  " and removeDate is null ";
10         //创建试图查询
11         String sql = "select
   readerid,readerName,readerTypeId,lendBookLimitted,createDate,creatorUserId,
   stopDate,stopUserId,readerTypeName" +
12                 " from view_reader" +
13                 " where removeDate is null ";
14         if (readerTypeId > 0) sql += " and readerTypeId=" + readerTypeId;
15         if (keyword != null && !"".equals(keyword))
16             sql += " and (readerid like ? or readerName like ?)";
17         sql += " order by readerid";
18         sql += "limit ?,?";
19         java.sql.PreparedStatement pst = conn.prepareStatement(sql);
20         if (keyword != null && !"".equals(keyword)) {
21             pst.setString(1, "%" + keyword + "%");
22             pst.setString(2, "%" + keyword + "%");
23             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;

    }
```

- ReaderManager.loadReader

```
public BeanReader loadReader(String readerid) throws DbException {
        Connection conn = null;
```

```
        try {
            conn = DBUtil.getConnection();
            //连接查询
//            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=?";
            //视图查询
            String sql = "select readerid,readerName,readerTypeId,lendBookLimitted,createDate,creatorUserId,stopDate,stopUserId,readerTypeName,removeDate" +
                    " from view_reader" +
                    " where 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));
                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;
    }
```

2、 设计图书视图view_book，并修改BookManager类中相关代码。

第一步：建立图书视图，要求视图中包含出版社名称；

第二步：改造BookManager类，将其中的连接查询用视图代替。

第三步：运行图书管理系统，进行各个功能的测试

【实验结果与分析】

A、 写出视图创建代码。

```
create view view_book as
select a.*, b.publishername
from beanbook a, beanpublisher b
where a.pubid = b.pubid;
```

B、 给出改造后BookManager类的各个方法的代码。

- BookManager.searchBook

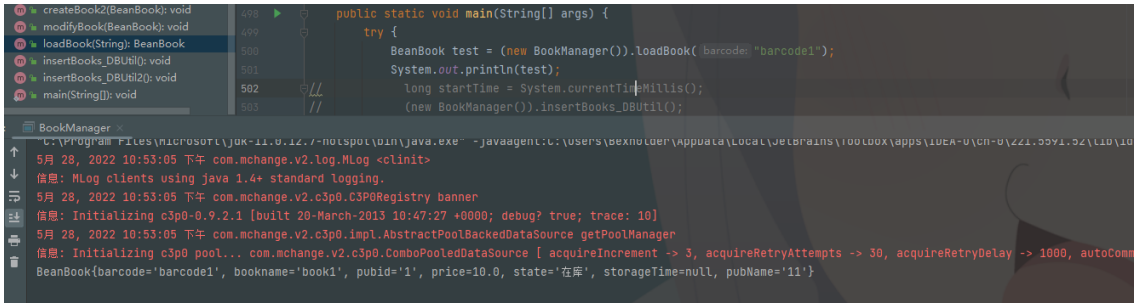
```
public List<BeanBook> searchBook(String keyword, String bookState) throws BaseException {
        List<BeanBook> result = new ArrayList<BeanBook>();
        Connection conn = null;
        try {
            conn = DBUtil.getConnection();
            //连接查询
//            String sql = "select b.barcode,b.bookname,b.pubid,b.price,b.state,p.publishername " +
//                    " from beanbook b left outer join beanpublisher p on (b.pubid=p.pubid)" +
//                    " where  b.state='" + bookState + "' ";
//            if (keyword != null && !"".equals(keyword))
//                sql += " and (b.bookname like ? or b.barcode like ?)";
//            sql += " order by b.barcode";
            //视图查询
```

```
14          String sql = "select barcode, bookname, pubid, price, state,
    publishername " +
15                  " from view_book" +
16                  " where  state='" + bookState + "' ";
17
18          if (keyword != null && !"".equals(keyword))
19              sql += " and (bookname like ? or barcode like ?)";
20          sql += " order by barcode";
21          java.sql.PreparedStatement pst = conn.prepareStatement(sql);
22          if (keyword != null && !"".equals(keyword)) {
23              pst.setString(1, "%" + keyword + "%");
24              pst.setString(2, "%" + keyword + "%");
25          }
26
27          java.sql.ResultSet rs = pst.executeQuery();
28          while (rs.next()) {
29              BeanBook b = new BeanBook();
30              b.setBarcode(rs.getString(1));
31              b.setBookname(rs.getString(2));
32              b.setPubid(rs.getString(3));
33              b.setPrice(rs.getDouble(4));
34              b.setState(rs.getString(5));
35              b.setPubName(rs.getString(6));
36              result.add(b);
37          }
38      } catch (SQLException e) {
39          e.printStackTrace();
40          throw new DbException(e);
41      } finally {
42          if (conn != null)
43              try {
44                  conn.close();
45              } catch (SQLException e) {
46                  // TODO Auto-generated catch block
47                  e.printStackTrace();
48              }
49      }
50      return result;
51
52  }
```

- BookManager.searchBook

```
public BeanBook loadBook(String barcode) throws DbException {
        Connection conn = null;
        try {
            conn = DBUtil2.getConnection();
//            String sql = "select
b.barcode,b.bookname,b.pubid,b.price,b.state,b.storagetime,p.publishername "
+ //修改处
//                    " from beanbook b left outer join beanpublisher p on
(b.pubid=p.pubid)" +
//                    " where  b.barcode=? ";
            String sql = "select barcode, bookname, pubid, price, state, storagetime,
publishername " + //修改处
                    " from view_book " +
                    " where barcode=? ";
            java.sql.PreparedStatement pst = conn.prepareStatement(sql);
            pst.setString(1, barcode);
            java.sql.ResultSet rs = pst.executeQuery();
            if (rs.next()) {
                BeanBook b = new BeanBook();
                b.setBarcode(rs.getString(1));
                b.setBookname(rs.getString(2));
                b.setPubid(rs.getString(3));
                b.setPrice(rs.getDouble(4));
                b.setState(rs.getString(5));
                b.setStorageTime(rs.getDate(6));//修改处
                b.setPubName(rs.getString(7));
                return b;
            }
        } catch (SQLException e) {
            e.printStackTrace();
            throw new DbException(e);
        } finally {
            if (conn != null)
                try {
                    conn.close();
```

```
32              } catch (SQLException e) {
33                  // TODO Auto-generated catch block
34                  e.printStackTrace();
35              }
36          }
37          return null;
38      }
```



3、 设计读者借阅情况统计视图**view_reader_static**，并在**BookLendManager**类中添加根据读者**ID**提取其借阅数量的代码。

第一步：建立读者统计视图，要求视图中包含读者ID、读者姓名、借阅数量；

第二步：在BookLendManager中添加方法 public int loadReaderLendCount(String readerid) throws DbException。并编写其代码

第三步：在BookLendManager类中添加main函数，并编写上述方法的测试代码。进行功能的测试

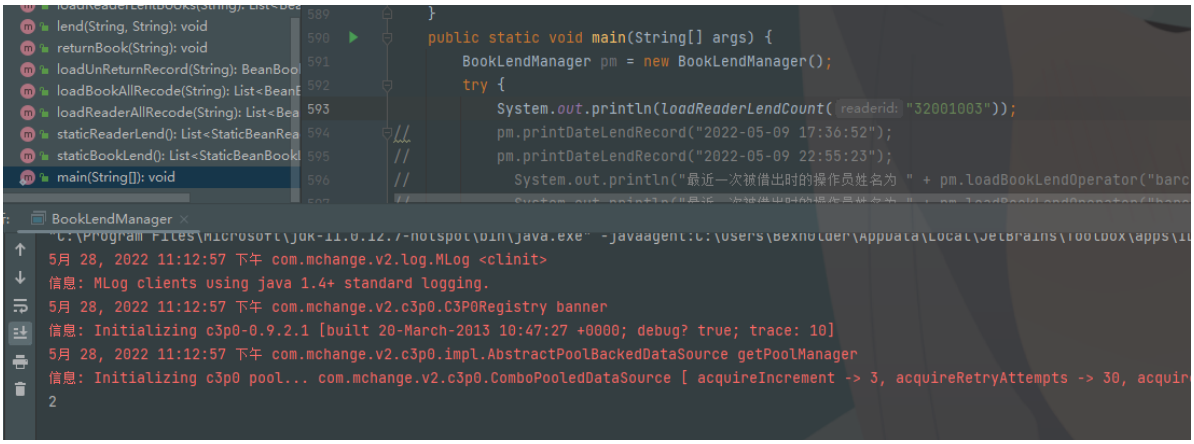
【实验结果与分析】

A、 写出视图创建代码。

```sql
create view view_reader_static as
select a.readerid, (count(*)-1) num
from (
    select readerid
    from beanreader
    union all
    select readerid
    from beanbooklendrecord
    where returnDate is null
) a
group by readerid;
```

B、 给出改造后BookLendManager类的各个方法的代码。

```java
public static int loadReaderLendCount(String readerid) throws Exception {
    Connection conn = null;
    PreparedStatement ps = null;
    ResultSet rs = null;
    try {
        conn = DBUtil2.getConnection();

        String sql = "select num from view_reader_static where readerid = ?";
        ps = conn.prepareStatement(sql);

        ps.setObject(1,readerid);

        rs = ps.executeQuery();

        if (rs.next()) {
            return rs.getInt(1);
        }
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        DBUtil2.closeResorce(conn,ps,rs);
    }

    return 0;
}
```



4、 索引实验：

第一步：完成IndexTest_initData类中的代码，并测试。

第二步：完成IndexTest类中的代码，并测试，记录执行结果

第三步：通过查询分析器，在BeanBookLendRecord表的readerid上建立索引

第四步：再次执行IndexTest类，记录执行结果

A、 给出两个类的代码和索引建立的代码

- IndexTest_initData

```
1  public void AutogenerationReaders(String readeridKeyword, String
   nameKeyword, int num, int PreReaderTypeId) throws BaseException{
2          Connection conn = null;
3          java.sql.PreparedStatement pst = null;
4          try {
5              conn = DBUtil.getConnection();
6              String sql = "select lendBookLimitted from BeanReaderType where
   readerTypeId=" + PreReaderTypeId;
7              java.sql.Statement st = conn.createStatement();
8              java.sql.ResultSet rs = st.executeQuery(sql);
9              if (!rs.next()) throw new BusinessException("读者类别不存在");
10             int lendBookLimitted = rs.getInt(1);
11             rs.close();
12             st.close();
13             sql = "insert into
   BeanReader(readerid,readerName,readerTypeId,lendBookLimitted,createDate,c
   reatorUserId) values(?,?,?,?,?,?)";
14             pst = conn.prepareStatement(sql);
15             //手动事务
16             conn.setAutoCommit(false);
```

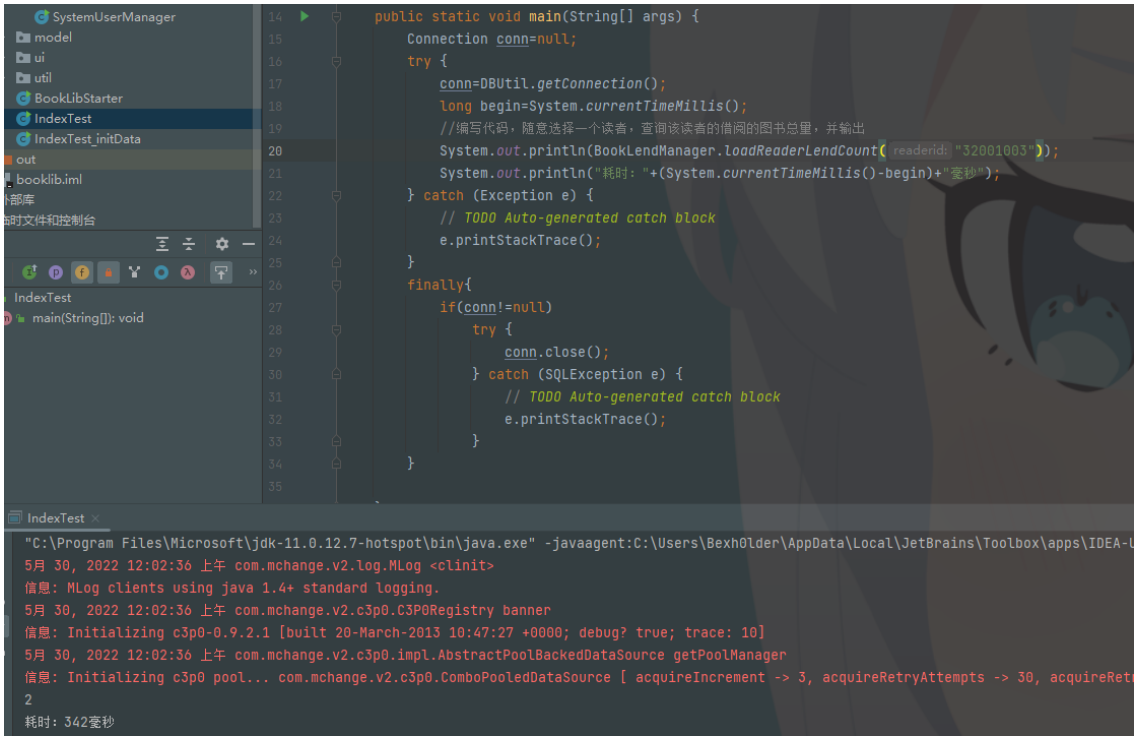
```
17            Long startTime = System.currentTimeMillis();
18            int count =0;
19            System.out.println("开始插入...");
20            for(int i = 0; i < num; i++){
21                pst.setString(1, readeridKeyword + String.valueOf(i));
22                pst.setString(2, nameKeyword + String.valueOf(i));
23                pst.setInt(3, PreReaderTypeId);//将所有创建的读者类型设为
24                pst.setInt(4, lendBookLimitted);
25                pst.setTimestamp(5, new
java.sql.Timestamp(System.currentTimeMillis()));
26                pst.setString(6, "admin"); //默认为admin创建
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                }
```

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

```
public void AutogenerationBooks(String barcodeKeyword, String bookNameKeyword,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.setString(1, barcodeKeyword + String.valueOf(i));
                pst.setString(2, bookNameKeyword + String.valueOf(i));
                pst.setString(3, Prepubid);
                pst.setDouble(4, 10);
                pst.setTimestamp(5, new java.sql.Timestamp(System.currentTimeMillis()));
                pst.addBatch();
                count++;
```

```
            if(count>=25000) {
                //每25000条数据进行一次批量插入操作
                pst.executeBatch();
                pst.clearBatch();
                conn.commit();
                count = 0;
            }
        }
        if(count != 0)
        {
            pst.executeBatch();
            pst.clearBatch();
            conn.commit();
            count = 0;
        }
        Long endTime = System.currentTimeMillis();
        System.out.println(num + "条数据插入完成,总用时： " + (endTime -
startTime)+"ms");
    } catch (Exception e) {
        e.printStackTrace();
        throw new RuntimeException(e);
    }finally{
        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);
            }
        }
    }
}
```

- IndexTest



B、 比较IndexTest类两次执行的结果，并说明索引的作用。并思考：如果我们需要经常查询某用户近期借阅的n本图书，请问，应该如何建立索引？【提示：多列索引】

```
ALTER TABLE `beanbooklendrecord` ADD INDEX
books_recently_borrowed_bytheuser(`readerid`, `lendDate`);
```

5、 数据统计实验：

A、在ReaderManager中，添加函数Map<String,Integer> staticTypeReaderCount()...

要求通过读者类别名称及其读者数量，请给出代码：

```
public Map<String,Integer> staticTypeReaderCount() throws Exception {
    Map<String, Integer> re = new HashMap<>();
```

```
3
4       Connection conn = DBUtil.getConnection();
5
6       String sql = "select readerTypeId, count(*) num from beanreader " + "group by
    readerTypeId";
7       java.sql.PreparedStatement ps = conn.prepareStatement(sql);
8
9       java.sql.ResultSet rs = ps.executeQuery();
10
11      while(rs.next()) {
12          String typeid = rs.getString(1);
13          int num = rs.getInt(2);
14          re.put(typeid,num);
15      }
16      System.out.println(re.size());
17      return re;
18    }
```

B、在BookManager中，添加函数Map<String,Double> staticPublisherBookAvgPrice()...。要求统计各出版社名称及其图书的平均价格，请给出代码：

```
1   public Map<String,Double> staticPublisherBookAvgPrice() throws SQLException {
2       Map<String, Double> re = new HashMap<>();
3
4       Connection conn = DBUtil.getConnection();
5
6       String sql = "select pubid, avg(price) avg_price from beanbook " + "group by
    pubid";
7       java.sql.PreparedStatement ps = conn.prepareStatement(sql);
8
9       java.sql.ResultSet rs = ps.executeQuery();
10
11      while(rs.next()) {
12          String typeid = rs.getString(1);
13          double price = rs.getInt(2);
14          re.put(typeid,price);
15      }
16      System.out.println(re.size());
17      return re;
18    }
```

C、在BookLendManager中，添加函数Map<String,Integer> staticReaderBookCount()... 。要求统计读者为归还的图书数量，返回结果的key为读者ID。请给出代码：

```
public Map<String,Integer> staticReaderBookCount() throws SQLException {
        Map<String, Integer> re = new HashMap<>();

        Connection conn = DBUtil.getConnection();

        String sql = "select readerid ,count(1) num" + " from beanbooklendrecord" + "
where returnDate is not null" + " group by readerid";

        java.sql.PreparedStatement ps = conn.prepareStatement(sql);

        java.sql.ResultSet rs = ps.executeQuery();

        while(rs.next()) {
            String typeid = rs.getString(1);
            int num = rs.getInt(2);
            re.put(typeid,num);
        }
        System.out.println(re.size());
        return re;
    }
```

6、 集合对象的遍历实验：

A、编写批量借阅读书函数： public void lendbooks(String readerId,Collection barcodes) .... 。其中第二个参数为图书条码集合。

```
public void lendbooks(String readerId,Collection<String> barcodes) throws
BaseException {
        Object[] objects = barcodes.toArray();
        for (int i = 0; i < objects.length; i++) {
            lend((String) objects[i],readerId);
        }
    }
```

B、编写批量设置罚金函数：public void setPenalSum(String readerId,Map<String,Double> penalSums) .... 。其中第二个参数的key为barcode，value为改读者尚未归还图书的罚金（注意，不要设置已经归还图书的罚金）。

```

```

```
1  public void setPenalSum(String readerId,Map<String,Double> penalSums) throws
   Exception {
2      Connection conn = null;
3      PreparedStatement ps = null;
4      try {
5          conn = DBUtil.getConnection();
6          String sql = "update beanbooklendrecord set penalSum = ? where readerid = ?
   and bookBarcode = ? and returnDate is null";
7          ps = conn.prepareStatement(sql);
8          ps.setObject(2,readerId);
9          for (Map.Entry<String,Double> x : penalSums.entrySet()) {
10             String barcode = x.getKey();
11             double penal = x.getValue();
12             ps.setObject(1,penal);
13             ps.setObject(3,barcode);
14             ps.execute();
15         }
16     } catch (SQLException e) {
17         e.printStackTrace();
18     } finally {
19         DBUtil.closeResorce(conn,ps);
20     }
21
22 }
```