附 录

附录一

1. 各类名词解释

1) 档案实体：档案实体在本文中专指纸质的档案文件

2) 档案状态：指房地产档案所处不同阶段的一种标识，用于标识档案生命周期中所处的阶段；本文中档案主要分为“未签收”、“签收”、“未校对”、“入库”、“出库”、“注销”六种状态。

3) 信息过滤标准数据表：该数据表用于存放具体功能编号所对应的标签过滤标准 XML 文件。具体格式如下：

表 1 信息过滤标准数据表

字段名称（列名）	中文解释	数据类型	长度
GNBH	功能编号	char	10
PATH	过滤标准位置	char	30
ZDR	制定人	char	20
ZDTIME	制定时间	long	10

4) 实体入库：新档案入库类似于通常档案管理中的录入功能，指一份新建要入库档案实体送至档案管后经过编目整理再由库房管理员根据指定位置放入库房的过程。旧档案的入库即是指将利用完的档案放回库房。

5) 实体出库：档案实体出库通常是指由申请利用的人员提出利用申请，再由库房管理员将批准利用的档案从库房取出交至利用者处。

6) 档案转架：库房管理中会因各类原因使档案需要存放入新的档案架，将档案由旧档案架放入新档案架的过程即称为档案转架。

附录二

1. 公共类：

1) 数据库访问类（DatabaseAccess）：

```
package commonClass.archives
```

```
import java.sql.*;
import java.io.*;
//提供JDBC与数据库之间的连接
public class DatabaseAccess{
    protected final String driver="sun.jdbc.odbc.JdbcOdbcDriver";
    protected String source="jdbc:odbc:";
    protected Connection connection;
    protected Statement statement;
    protected PreparedStatement prepared;
    //构造方法，建立与数据库的连接
    public DatabaseAccess(String source)throws SQLException{
    try{//查找用于JDBC驱动的类，这种查找可向驱动器管理器注册该数据库驱动程序
    this.source=this.source+source;
    Class.forName(driver);
    }catch(ClassNotFoundException exc){
    //当没有驱动程序时，应用程序无法继续运行，故退出程序
    System.out.printIn("没有发现驱动程序；"+driver);
    exc.printStackTrace();System.exit(1);
    }
    //建立与数据库的连接
    this.source=source;
    connection=DriverManager.getConnection(source);
    //如果连接成功则检测是否有警告信息
    SQLWarning warn=connection.getWarnings();
    while(warn!=null){
    System.out.printIn(warn.getMessage());
    warn=warn.getNextWarning)();}
    //创建一个用于执行简单SQL语句对象
    statement=connection.createStatement();}
    //关闭数据库的连接
```

```
public void close()throws SQLException{
//关闭连接
if(prepared!=null)prepared.close();
if(connection!=null)connection.commit();
if(connection!=null)prepared.close();
}
//利用一条 SQL 语句执行数据库查询操作。参数 sql 表示 SQL 查询语句串
（如”SELECT *FROM user”）,返回查询结果集
Public RestultSet query(String sql)throws SQLException{
ResultSet rs=statement.executeQuery(sql);
Return rs;
//利用一条 SQL 语句执行数据库更新操作。参数 sql 表示 SQL 更新语句串
public void update(String sql)throws SQLException{
    statement.executeUpdate(sql);
    }
    //以预编译 SQL 语句查询与更新数据库的方法准备一条预编译的 SQL 语句。
    //参数 sql 表示执行查询或更新的 SQL 语句
    public void prepare(String sql)throws SQLException{
        prepared=connection.prepareStatement(sql);}
//执行以编译好的 SQL 语句以完成数据库查询操作。返回查找结果
    public ResultSet preparedQuery() throws SQLException{
        ResultSet result=prepared.executeQuery();
        return result;}
            ..................//其余代码省略}
```

2) 标签类

```
package commonClass.archives
pulic class Tag{
    public int blockNum;//欲读取的标签中数据块数
    public String[] field;//标签中数据块名组成的字符串
    public int[] start;//标签中数据块开始位置组成的整数串
```

```
public int[] length;//标签中数据块长度组成的整数串
public Tag(){//
    }
 //read 方法从标签中读出数据值
 public String[] read(int blockNum){
        //以长度为 blockNum 的字符串数组存储并返回电子标签中数据块的值
        String[] value=new String[blockNum];
       //以下实现向读写器发送指令读取电子标签中特定位置数据块的值，代码省略
        ..........
        return value;
        }
//write 方法实现按照指定格式向标签写入数据块值操作
public void write(String[] value,int blockNum){
            //欲写出的值串个数小于应写入的数据块个数，则剩余值为空字符串。
      if(value.length<blockNum){
            String[] otherValue=new String[blockNum-value.length];
      for(int i=0;i<blockNum-value.length;i++)otherValue[i]=" ";}
      //以下代码实现向电子标签写入 blockNum 个特定位置数据块值，代码省略..
      .......
            }
      }
```

3) 标签数据过滤标准类

```
package commonClass.archives;
import javax.xml.parsers.FactoryConfigurationError;
import javax.xml.parsers.ParserConfigurationException;
import java.io.IOException;
import java.sql.ResultSet;
import java.sql.Connection;
import java.sql.PreparedStatement;
public class FiltrateStandard{
```

```
private String XMLPath;
public String DBName;
public String functionID;
private int num;
//
public FiltrateStandard(String DBName,String functionID){
    this.DBName=DBName;
    this.functionID=functionID;
}
public int getBlockNum(){return num;}
//根据特定功能ID获取载有数据过滤标准XML文件的路径
private String getFiltrateFile(){
      if(functionID.equals(" "))return " ";
      try{
      DatabaseAccess    dbAccess=new DatabaseAccess(DBName);
      String sql=" SELECT filePath from XXGLBZSJB where functionID= " +functionID;
      ResultSet   midres=dbAccess.query(sql);
      while(midres.next()){
      String   res=midres.getString(1);
      dbAccess.close();}
      }catch(SQLException exc){
          exc.printStackTrace();System.exit(1);}
      //getStandard()方法返回格式为"field-start-length"，例如在分析完档案签收
    //的过滤标准文件后，得出一组字符串值，"QSLSH-1-20"表明从可读写电子标签位置
    //1起20个单位长度的数据表示的是QSLSH。
      public String[] getStandard(){
      XMLPath= getFiltrateFile();
      if(XMLPath.equals(" "))return null;
      try{
      XMLTagManager   mana=XMLTagManager(XMLPath);
```

```
        return mana. TagFiltrateReader(); }
    catch(Exception e){
            e.printStackTrace(); }
            }
}
```

2. 条形码系统与 RFID 系统共用表

表 1 案件信息表

字段名称（列名）	中文解释	数据类型	长度
YWZG	业务字轨	char	10
YWAH	业务案号	char	10
YWBS1	其他业务标识 1	char	20
YWBS2	其他业务标识 2	char	20
QSLSH	档案签收流水号	char	20
PROOFPERSON	档案整理人	char	20
PROOFTIME	整理时间	long	10
DAZT	档案状态	char	10

表 2 档案信息表

字段名称（列名）	中文解释	数据类型	长度
QSLSH	档案签收流水号	char	20
YWZG	业务字轨	char	10
YWAH	业务案号	char	10
YWBS1	其他业务标识 1	char	20
YWBS2	其他业务标识 2	char	20
GDH	归档号	char	20
GDTYPE1	档案类型 1	char	20
GDTYPE2	档案类型 2	char	20
LAADDRESS	立案地址	char	50

TH	图号	char	20
FH	幅号	char	20
QH	区号	char	20
DH	段号	char	20
QDDH	区段地号	char	20
DZGDZT	电子归档状态	boolean	
STZKZT	实体在库状态	char	10
LOGOUTFZ	注销附注	char	30
LOGOUTP	注销人	char	20
LOGOUTT	注销时间	long	10
LOGZT	记录状态	char	10

表 3 盘点表

字段名称（列名）	中文解释	数据类型	长度
CHECKLSH	盘点流水号	char	20
CHECKTYPE	盘点类型	char	10
CHECKFIELD	盘点范围	char	40
CHECKTIME	盘点日期	long	10
CHECKPERSON	盘点人	char	20
REMARK	备注	char	50