

**本科生课程设计说明书**

**( 作品说明书 )**

**信息工程学院**

**题 目： 图书管理系统**

**学 号： \*\*\*\*\*\*\*\*\*\*\*\*\***

**姓 名： \*\*\*\*\***

**专业班级： 计算机\*\*\*\*\*班**

**2018年6月9日**

# 

目录

[1 项目背景及意义 1](#_Toc12585)

[2 需求分析 1](#_Toc29961)

[2.1 现状分析 1](#_Toc5779)

[2.2 系统功能需求 1](#_Toc21476)

[2.3 系统设计目标 2](#_Toc14809)

[3 系统可行性分析 2](#_Toc25391)

[3.1 技术可行性 2](#_Toc30393)

[3.2 经济可行性 2](#_Toc27112)

[3.3 操作可行性 2](#_Toc12714)

[4 使用的相关技术 3](#_Toc13927)

[4.1 C# 3](#_Toc5208)

[4.2 SQL server 3](#_Toc30074)

[5 系统详细设计 3](#_Toc25318)

[5.1 总体设计 3](#_Toc8986)

[5.2 数据库设计 3](#_Toc22593)

[5.2.1 数据流图 3](#_Toc27125)

[5.2.2 数据字典 4](#_Toc10763)

[5.2.3 逻辑结构 6](#_Toc12328)

[5.3 安全性设计 7](#_Toc6375)

[5.4 系统功能设计 7](#_Toc17110)

[6 方案分析 8](#_Toc22579)

[6.1 可操作性分析 8](#_Toc24458)

[6.2 先进性分析 8](#_Toc12398)

[6.3 稳健性分析 8](#_Toc4476)

[6.4 可扩展性分析 8](#_Toc6612)

[6.5 可维护性分析 8](#_Toc18998)

[7 系统测试 9](#_Toc14596)

[7.1 测试概述 9](#_Toc6188)

[7.2 测试资源与环境 9](#_Toc13737)

[7.2.1 硬件配置 9](#_Toc13122)

[7.2.2 软件配置 9](#_Toc25460)

[7.3 测试内容 9](#_Toc5720)

[7.3.1 管理员模块 9](#_Toc14320)

[7.3.2 图书管理模块 10](#_Toc22093)

[7.3.3 用户管理模块 11](#_Toc7160)

[7.3.4 图书借还模块 12](#_Toc10785)

[7.3.5 分类管理模块 12](#_Toc24022)

[7.3.6 查询模块 13](#_Toc29979)

[8 心得体会 14](#_Toc29007)

[9 附录 15](#_Toc27418)

[9.1 部分源码 15](#_Toc21808)

[9.1.1 图书借还 15](#_Toc26751)

[9.2.2 图书管理及查询 18](#_Toc29200)

# 1 项目背景及意义

当今由于信息技术的飞速发展，图书馆作为社会知识信息媒介的功能日益重要，网络环境下的信息资源建设知识仓库的设计，开放存取学术交流模式，知识管理系统，智能检索，数字参考咨询，数字图书馆领域成为图书馆系统的研究方向。

管理发展至今，综合发展的整体趋势已日渐明显。引进新方法，吸收新思想是促进图书馆系统发展的必要条件。因此，图书馆系统的研究要坚持理论与技术相融合，开展跨学科的交叉研究；要坚持理论与实践相结合，而对现实，回答现实所提出的最迫切的问题；要坚持基础研究和应用研究，前者对后者起指导作用，后者反过来对前者起丰富和发展的作用；要正确处理现实、历史、理论三者之间的关系，这是推进管理系统研究的重要方法。

# 2 需求分析

## 2.1 现状分析

以往，人们使用传统的人工方式管理图书馆的日常业务，其操作流程比较繁琐。在借书时，读者首先将要借的书好借阅证交给工作人员，然后工作人员将每本书的信息卡片好读者的借阅证放在一个小格栏里，最后在借阅证和每本书贴的借阅条上填写借阅信息。在还书时，读者首先将要还的书交给工作人员，然后工作人员根据图书信息找到相应的书卡好借阅证，并填写相应的还书信息。

学校图书馆有许多书，如果靠人工的方式来进行管理，那么工作量会非常大。图书的统计和管理会成为让人头疼的问题，读者借阅图书进行登记也会十分的不方便。这时，利用计算机来为我们进行这些统计管理的工作，可以给我们的生活带来大大的便利，并且提高工作效率。由于条件有限，只能使用键盘输入书的信息，所以在输入图书的ISBN号时有些麻烦。所以在课设里，采用了图书编号。在现实生活中，采用扫描的方式录入ISBN号会更加准确，高效。

## 

## 2.2 系统功能需求

图书管理系统是针对图书馆工作人员开发的管理软件，实现对图书信息、用户信息的管理功能。

本系统主要能够完成以下功能：

1.图书信息管理（查询、修改、新增、删除）

2.用户信息管理（查询、修改、新增、删除）

3.借书操作、还书操作、续借操作、借书记录查看

4.图书和用户分类管理（查询、修改、新增、删除）

## 

## 2.3 系统设计目标

1.系统中的数据表设计合理、高效、数据冗余少

2.软件界面友好、安全性高

3.易于维护、方便升级

# 3 系统可行性分析

## 3.1 技术可行性

图书管理系统的开发环境是Visual Studio 2010、Microsoft SQL Server 2008R2 SP3。系统是应用C/S的交互方式实现，本系统具有稳定性、安全性、美观性、先进性。系统现阶段的发展过程中，利用现有的人力物力是完全具备的能力开发出来的，作为阶段性产品，日后的发展空间大，实现方法逐步简单容易，所以图书管理系统的技术上是完全可行的。在当前的限制条件下，该系统的功能目标能达到。

## 3.2 经济可行性

计算机专业师生团队可以进行日常的系统维护，工作量少，经济支出少。

## 

## 3.3 操作可行性

所有界面具有风格统一，不同栏目的页面文字风格统一，重点突出图书管理平台的特点，系统页面友好，操作简单，即使没有计算机基础的老师也很好上手。

# 4 使用的相关技术

## 4.1 C#

C#是微软公司发布的一种面向对象的、运行于.NET Framework之上的高级程序设计语言。并定于在微软职业开发者论坛（PDC）上登台亮相。C#是一种安全的、稳定的、简单的、优雅的，由C和C++衍生出来的面向对象的编程语言。它在继承C和C++强大功能的同时去掉了一些它们的复杂特性（例如没有宏以及不允许多重继承）。C#综合了VB简单的可视化操作和C++的高运行效率，以其强大的操作能力、优雅的语法风格、创新的语言特性和便捷的面向组件编程的支持成为.NET开发的首选语言。

## 4.2 SQL server

SQL Server是由Microsoft开发和推广的关系数据库管理系统（DBMS），它最初是由Microsoft、Sybase和Ashton-Tate三家公司共同开发的，并于1988年推出了第一个OS/2版本。Microsoft SQL Server近年来不断更新版本，1996年，Microsoft 推出了SQL Server 6.5版本；1998年，SQL Server 7.0版本和用户见面；SQL Server 2000是Microsoft公司于2000年推出，目前最新版本是2017年份推出的SQL SERVER 2017。

# 5 系统详细设计

## 5.1 总体设计

以Microsoft SQL Server 2008R2 sp3、Visual Studio 2010作为开发环境，使用数据库技术以及.net开发技术，从用C#三层架构：表现层（UI（User Interface））、业务逻辑层（BLL（Business Logic Layer））、数据访问层（DAL（Data Access Layer））再加上实体类库（Model），开发此图书管理系统。

## 

## 5.2 数据库设计

### 5.2.1 数据流图

图书借还信息表

用户信息管理

图书信息管理

图书信息表

用户信息表

图书借还管理

图书管理系统

图书分类表

用户学院表

管理员表

管理员管理

分类管理

用户类型表

用户班级表

图5.1 数据流图

### 5.2.2 数据字典



图5.2 Admin表



图5.3 BookInfo表



图5.4 BookType表



图5.5 BorrowReturn表



图5.6 Class表



图5.7 Department表



图5.8 Reader表



图5.9 ReaderType表

### 5.2.3 逻辑结构

管理员（**管理员ID**，密码，类型，备注）

图书信息（**图书ID**，书名，入库时间，作者，拼音码，翻译者，语言，页数，价格，印刷版面，存放位置，ISBN码，版本，备注）

图书类型（**图书类型ID**，图书类型名称）

图书借还（**图书借还ID**，图书ID，用户ID，借书时间，应还书时间，真实还书时间，罚金，续借次数，备注）

用户班级（**班级ID**，班级名称）

用户学院（**学院ID**，学院名称）

用户信息（**用户ID**，用户姓名，登记时间，有效期，用户类型，学院，班级，身份证号，性别，QQ，电话，Email，地址，备注）

用户学院（**用户类型ID**，用户类型名称）

## 5.3 安全性设计

系统有两种管理员：第一种是普通管理员，拥有除管理员管理之外所以权限；第二种是超级管理员，拥有最高权限，可以修改普通管理员。这样的设定使得数据库数据不会被随意篡改。

## 5.4 系统功能设计

图5.10 系统概念设计图

# 6 方案分析

## 6.1 可操作性分析

所有页面具有统一的风格，页面文字风格统一，在构思上重点突出图书管理平台的特点，以获得较佳的浏览效率。

## 

## 6.2 先进性分析

本系统是应用C/S的交互方式实现，从用C#三层架构：表现层（UI（User Interface））、业务逻辑层（BLL（Business Logic Layer））、数据访问层（DAL（Data Access Layer））再加上实体类库（Model），开发此图书管理系统，达到了本系统的稳定性、安全性、美观性性、先进性。

## 

## 6.3 稳健性分析

系统目前还处在完成最基本功能阶段，不会有大量的用户访问，服务器使用阿里云，稳健性较好。

## 

## 6.4 可扩展性分析

系统页面简洁，添加新页面很方便，新功能并不会与已有功能相冲突，具有很强的扩展性。而且系统中的数据库可以随学校的规模的变化进行动态的扩充或缩减。

## 

## 6.5 可维护性分析

程序简单，数据库数据冗余少，使用日志文件等可恢复数据库，可维护性强。

# 

# 7 系统测试

## 7.1 测试概述

编写测试用例并进行测试，找出bug并改正。

## 

## 7.2 测试资源与环境

### 7.2.1 硬件配置

电脑型号：战神Z6S1

处理器：Intel（R）Core(TM)i7-4720HQ CPU @2.60GHz 2.60GHz

### 7.2.2 软件配置

操作系统：Windows 10 专业版 64位操作系统

使用软件：Microsoft Visual Studio 2010

SQL server 2008R2 SP3

## 

## 7.3 测试内容

### 7.3.1 管理员模块



图7.1 管理员模块

### 7.3.2 图书管理模块



图7.2 图书管理模块



图7.2 新增图书

### 7.3.3 用户管理模块



图7.4 用户管理模块

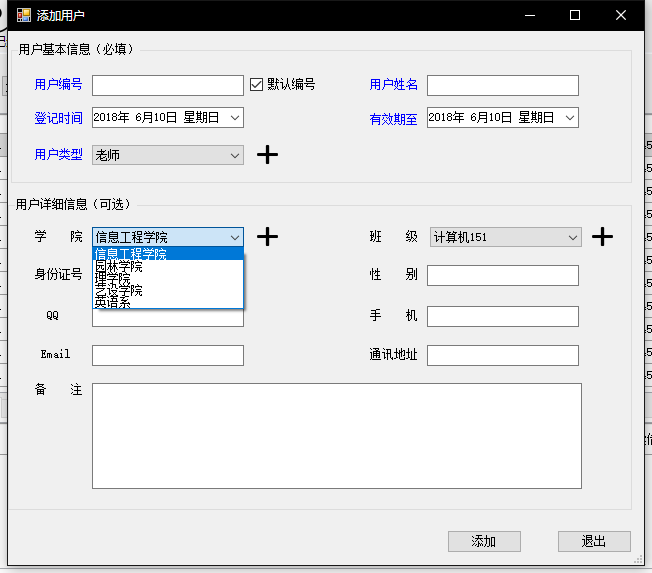


图7.5 用户新增

### 7.3.4 图书借还模块

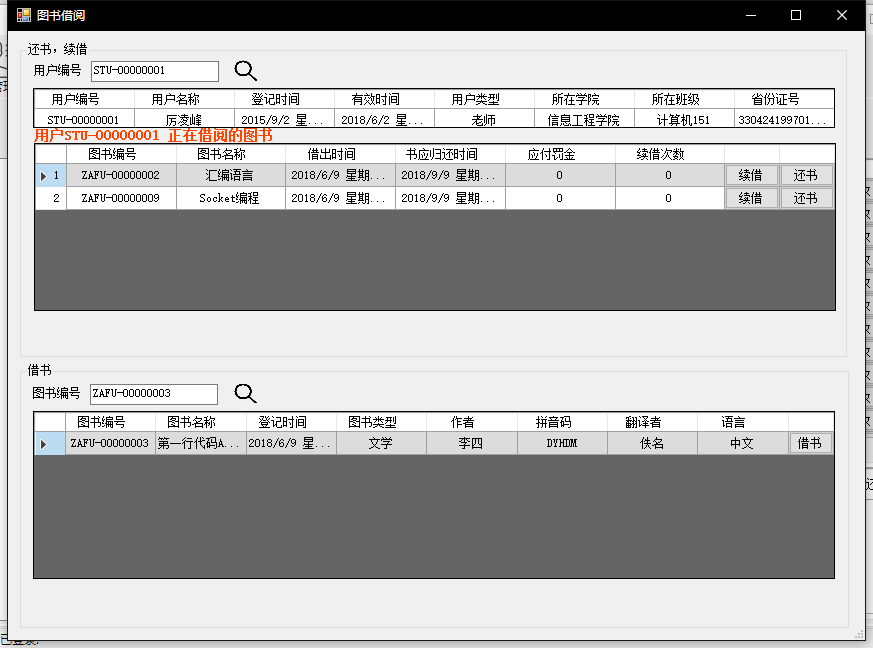


图7.6 图书借还

### 7.3.5 分类管理模块

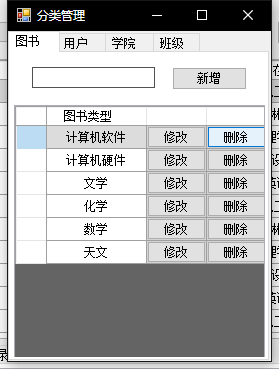


图7.7 分类管理模块

### 7.3.6 查询模块

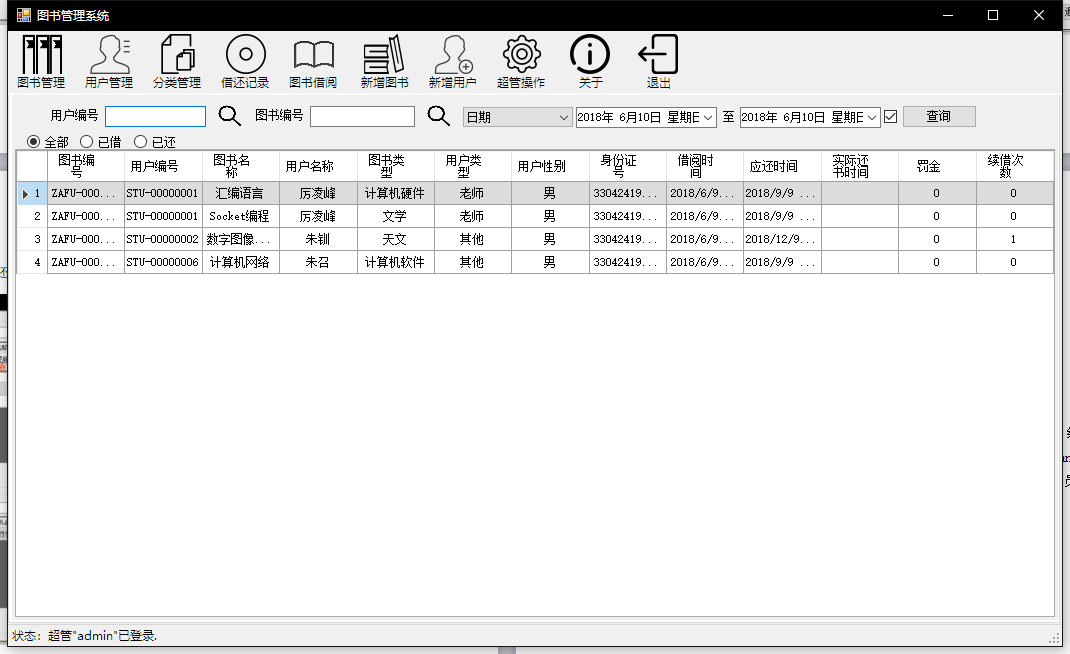


图7.8 图书借还记录



图7.9 查询

# 8 心得体会

通过本次课设，复习了一个学期的学习内容，使用了c#三层架构去写程序，结构更加清晰，效率更高。数据库连接语句放在App.Config中，数据库操作语句采用Parameter，方便日后修改维护，在数据库的设计上满足了第三范式。不过由于时间原因，管理员修改还没有完成。

感谢老师一学期的教导，祝老师生活愉快！

# 9 附录

## 9.1 部分源码

### 9.1.1 图书借还

//查询BorrowReturn表信息

public DataSet selectHostory(BorrowReturn b, string radioName, String cboBorrowTimeType, Boolean checkTime)

{

string sql = string.Format(@"select BookInfo.BookId as 'BookId',Reader.UserId as 'UserId',BookName,UserName,BookTypeName,UserTypeName,Gender,IdentityCard,BorrowTime,ReturnTime,FactReturnTime,Fine,RenewCount from BorrowReturn

inner join BookInfo on BookInfo.BookId=BorrowReturn.BookId

inner join Reader on Reader.UserId=BorrowReturn.UserId

inner join BookType on BookType.BookTypeId=BookInfo.BookTypeId

inner join ReaderType on ReaderType.UserTypeId=Reader.UserTypeId

where BookInfo.BookId like '%{0}%' and

Reader.UserId like '%{1}%' ", b.BookId, b.UserId);

if (radioName == "全部")

{

}

else if (radioName == "已借")

{

sql += " and FactReturnTime is null ";

}

else if (radioName == "已还")

{

sql += " and FactReturnTime is not null ";

}

return DBhelp.Create().ExecuteAdater(sql);

}

//查询BorrowReturn表全部信息

public DataSet AllBorrowReturn()

{

string sql = @"select BookInfo.BookId as 'BookId',Reader.UserId as 'UserId',BookName,UserName,BookTypeName,UserTypeName,Gender,IdentityCard,BorrowTime,ReturnTime,FactReturnTime,Fine,RenewCount from BorrowReturn

inner join BookInfo on BookInfo.BookId=BorrowReturn.BookId

inner join Reader on Reader.UserId=BorrowReturn.UserId

inner join BookType on BookType.BookTypeId=BookInfo.BookTypeId

inner join ReaderType on ReaderType.UserTypeId=Reader.UserTypeId";

return DBhelp.Create().ExecuteAdater(sql);

}

//查询图书BorrowReturn表（表连接）

public DataSet selectBorrowReturn(string BookId)

{

string sql = @"select BookInfo.BookId,BookName,Reader.UserId,UserName,BorrowTime,ReturnTime,FactReturnTime,Fine,RenewCount,BorrowRemark from BookInfo

inner join BorrowReturn on BorrowReturn.BookId=BookInfo.BookId

inner join Reader on Reader.UserId=BorrowReturn.UserId

where BookInfo.BookId=@BookId ";

SqlParameter[] sp ={

new SqlParameter("@BookId",BookId)

};

return DBhelp.Create().ExecuteAdater(sql, sp);

}

//用户借阅记录

public DataSet ReaderBorrowReturn(string UserId)

{

string sql = @"select BorrowId,Reader.UserId as 'UserId',UserName,BookInfo.BookId as 'BookId',BookName,BorrowTime,ReturnTime,FactReturnTime,Fine,RenewCount,BorrowRemark from Reader

inner join BorrowReturn on BorrowReturn.UserId=Reader.UserId

inner join BookInfo on BookInfo.BookId=BorrowReturn.BookId

where Reader.UserId=@UserId and FactReturnTime is null ";

SqlParameter[] sp ={

new SqlParameter("@UserId",UserId)

};

return DBhelp.Create().ExecuteAdater(sql, sp);

}

//用户历史借阅记录

public DataSet ReaderBorrowReturn1(string UserId)

{

string sql = @"select BorrowId,Reader.UserId as 'UserId',UserName,BookInfo.BookId as 'BookId',BookName,BorrowTime,ReturnTime,FactReturnTime,Fine,RenewCount,BorrowRemark from Reader

inner join BorrowReturn on BorrowReturn.UserId=Reader.UserId

inner join BookInfo on BookInfo.BookId=BorrowReturn.BookId

where Reader.UserId=@UserId and FactReturnTime is not null ";

SqlParameter[] sp ={

new SqlParameter("@UserId",UserId)

};

return DBhelp.Create().ExecuteAdater(sql, sp);

}

//还书

public int ReturnBook(int BorrowReturnId)

{

string sql = @"update BorrowReturn set FactReturnTime=@FactReturnTime,RenewCount=0 where BorrowId=@BorrowId";

SqlParameter[] sp ={

new SqlParameter("@FactReturnTime",DateTime.Now),

new SqlParameter("@BorrowId",BorrowReturnId)

};

return DBhelp.Create().ExecuteNonQuery(sql, sp: sp);

}

//借书

public int BorrowBook(BorrowReturn b)

{

string sql = "proc\_BorrowBook";

Console.WriteLine(b.BookId);

SqlParameter[] sp ={

new SqlParameter("@BorrowId",DbType.Int32),

new SqlParameter("@BookId",b.BookId),

new SqlParameter("@UserId",b.UserId),

new SqlParameter("@BorrowTime",b.BorrowTime),

new SqlParameter("@ReturnTime",b.ReturnTime),

new SqlParameter("@Fine",b.Fine),

new SqlParameter("@RenewCount",b.RenewCount),

new SqlParameter("@BorrowRemark",b.BorrowRemark),

new SqlParameter("@ReturnValue",DbType.Int32)

};

sp[0].Direction = ParameterDirection.Output;

sp[sp.Length - 1].Direction = ParameterDirection.ReturnValue;

DBhelp.Create().ExecuteNonQuery(sql, CommandType.StoredProcedure, sp);

b.BorrowId = (int)sp[0].Value;

return (int)sp[sp.Length - 1].Value;

}

//续借

public int RenewBook(BorrowReturn b)

{

string sql = @"update BorrowReturn set ReturnTime=dateadd(month,3,ReturnTime),RenewCount=RenewCount+1 where BorrowId=@BorrowId";

SqlParameter[] sp ={

new SqlParameter("@BorrowId",b.BorrowId)

};

return DBhelp.Create().ExecuteNonQuery(sql, sp: sp);

}

### 9.2.2 图书管理及查询

//查询BookInfo表

public List<BookInfo> selectBookInfo()

{

string sql = @"select BookId,BookName,TimeIn,BookTypeName,Author,PinYinCode,Translator,Language,BookNumber,Price,Layout,Address,ISBN,Versions,BookRemark from BookInfo

inner join BookType on BookType.BookTypeId=BookInfo.BookTypeId";

List<BookInfo> list = new List<BookInfo>();

SqlDataReader reader = DBhelp.Create().ExecuteReader(sql);

while (reader.Read())

{

BookInfo b = new BookInfo();

b.BookId = reader.GetString(0);

b.BookName = reader.GetString(1);

b.TimeIn = reader.GetDateTime(2);

b.BookType = new BookType();

b.BookType.BookTypeName = reader.GetString(3);

b.Author = reader.GetString(4);

b.PinYinCode = reader.GetString(5);

b.Translator = reader.GetString(6);

b.Language = reader.GetString(7);

b.BookNumber = reader.GetString(8);

b.Price = reader.GetString(9);

b.Layout = reader.GetString(10);

b.Address = reader.GetString(11);

b.ISBN = reader.GetString(12);

b.Versions = reader.GetString(13);

b.BookRemark = reader.GetString(14);

list.Add(b);

}

reader.Close();

return list;

}

//根据ID查询bookinfo表

public List<BookInfo> selectBookInfo(string BookId)

{

string sql = @"select BookId,BookName,TimeIn,BookTypeId,Author,PinYinCode,Translator,Language,BookNumber, Price,Layout,Address,ISBN,Versions,BookRemark from BookInfo where BookId=@BookId";

SqlParameter[] sp ={

new SqlParameter("@BookId",BookId)

};

SqlDataReader reader = DBhelp.Create().ExecuteReader(sql, sp);

List<BookInfo> list = new List<BookInfo>();

while (reader.Read())

{

BookInfo b = new BookInfo();

b.BookId = reader.GetString(0);

b.BookName = reader.GetString(1);

b.TimeIn = reader.GetDateTime(2);

b.BookTypeId = reader.GetInt32(3);

b.Author = reader.GetString(4);

b.PinYinCode = reader.GetString(5);

b.Translator = reader.GetString(6);

b.Language = reader.GetString(7);

b.BookNumber = reader.GetString(8);

b.Price = reader.GetString(9);

b.Layout = reader.GetString(10);

b.Address = reader.GetString(11);

b.ISBN = reader.GetString(12);

b.Versions = reader.GetString(13);

b.BookRemark = reader.GetString(14);

list.Add(b);

}

reader.Close();

return list;

}

//查询BookInfo表

public DataSet selectBookInfo1()

{

string sql = @"select BookId,BookName,TimeIn,BookTypeName,Author,PinYinCode,Translator,Language,BookNumber,Price,Layout,Address,ISBN,Versions,BookRemark from BookInfo inner join BookType on BookType.BookTypeId=BookInfo.BookTypeId";

return DBhelp.Create().ExecuteAdater(sql);

}

public DataSet selectBook;

//查询BookInfo表 目前可以借阅的图书

public DataSet selectBookInfo2(string BookId)

{

string sql = @"select BookInfo.BookId as 'BookId',BookName,TimeIn,BookTypeName,Author,PinYinCode,Translator,Language,BookNumber,Price,Layout,Address,ISBN,Versions,BookRemark from BookInfo

inner join BookType on BookType.BookTypeId=BookInfo.BookTypeId

where BookInfo.BookId like '%'+@BookId+'%' and BookInfo.BookId not in(select BookId from BorrowReturn where FactReturnTime is null )";

SqlParameter[] sp ={

new SqlParameter("@BookId",BookId)

};

return DBhelp.Create().ExecuteAdater(sql, sp: sp);

}

//根据条件查询bookinfo表

public DataSet selectBookInfo1(int index)

{

string sql = @"select BookId,BookName,TimeIn,BookTypeName,Author,PinYinCode,Translator,Language,BookNumber,Price,Layout,Address,ISBN,Versions,BookRemark from BookInfo

inner join BookType on BookType.BookTypeId=BookInfo.BookTypeId

where BookType.BookTypeId=@BookTypeId";

SqlParameter[] sp ={

new SqlParameter("@BookTypeId",index)

};

return DBhelp.Create().ExecuteAdater(sql, sp);

}

//根据条件查询bookinfo表

public DataSet selectBookInfo1(string A, string B)

{

string sql = string.Format(@"select BookId,BookName,TimeIn,BookTypeName,Author,PinYinCode,Translator,Language,BookNumber,Price,Layout,Address,ISBN,Versions,BookRemark from BookInfo

inner join BookType on BookType.BookTypeId=BookInfo.BookTypeId

where {0} like '%{1}%'", A, B);

return DBhelp.Create().ExecuteAdater(sql);

}

//查询BookInfo表 带全部查询(表中所有相关的字段)

public DataSet selectBookInfo1(List<string> list, string B)

{

string sql = "";

for (int i = 0; i < list.Count; i++)

{

if (i != list.Count - 1)

{

sql += string.Format(@"select BookId,BookName,TimeIn,BookTypeName,Author,PinYinCode,Translator,Language,BookNumber,Price,Layout,Address,ISBN,Versions,BookRemark from BookInfo

inner join BookType on BookType.BookTypeId=BookInfo.BookTypeId

where {0} like '%{1}%' union ", list[i], B);

}

else

{

sql += string.Format(@"select BookId,BookName,TimeIn,BookTypeName,Author,PinYinCode,Translator,Language,BookNumber,Price,Layout,Address,ISBN,Versions,BookRemark from BookInfo

inner join BookType on BookType.BookTypeId=BookInfo.BookTypeId

where {0} like '%{1}%' ", list[i], B);

}

}

return DBhelp.Create().ExecuteAdater(sql);

}

//添加图书信息

public int AddBookInfo(BookInfo book)

{

string sql = @"insert into BookInfo select @BookId,@BookName,@TimeIn,@BookTypeId,@Author,@PinYinCode,@Translator,@Language,@BookNumber,@Price,@Layout,@Address,@ISBN,@Versions,@BookRemark";

SqlParameter[] sp ={

new SqlParameter("@BookId",book.BookId),

new SqlParameter("@BookName",book.BookName),

new SqlParameter("@TimeIn",book.TimeIn),

new SqlParameter("@BookTypeId",book.BookTypeId),

new SqlParameter("@Author",book.Author),

new SqlParameter("@PinYinCode",book.PinYinCode),

new SqlParameter("@Translator",book.Translator),

new SqlParameter("@Language",book.Language),

new SqlParameter("@BookNumber",book.BookNumber),

new SqlParameter("@Price",book.Price),

new SqlParameter("@Layout",book.Layout),

new SqlParameter("@Address",book.Address),

new SqlParameter("@ISBN",book.ISBN),

new SqlParameter("@Versions",book.Versions),

new SqlParameter("@BookRemark",book.BookRemark),

};

return DBhelp.Create().ExecuteNonQuery(sql, sp: sp);

}

//修改图书信息

public int ExitBookInfo(BookInfo book)

{

string sql = @"update BookInfo set BookName=@BookName,TimeIn=@TimeIn,BookTypeId=@BookTypeId, Author=@Author,PinYinCode=@PinYinCode,Translator=@Translator,Language=@Language,BookNumber=@BookNumber,Price=@Price,Layout=@Layout,Address=@Address,ISBN=@ISBN,Versions=@Versions,BookRemark=@BookRemark

where BookId=@BookId";

SqlParameter[] sp ={

new SqlParameter("@BookName",book.BookName),

new SqlParameter("@TimeIn",book.TimeIn),

new SqlParameter("@BookTypeId",book.BookTypeId),

new SqlParameter("@Author",book.Author),

new SqlParameter("@PinYinCode",book.PinYinCode),

new SqlParameter("@Translator",book.Translator),

new SqlParameter("@Language",book.Language),

new SqlParameter("@BookNumber",book.BookNumber),

new SqlParameter("@Price",book.Price),

new SqlParameter("@Layout",book.Layout),

new SqlParameter("@Address",book.Address),

new SqlParameter("@ISBN",book.ISBN),

new SqlParameter("@Versions",book.Versions),

new SqlParameter("@BookRemark",book.BookRemark),

new SqlParameter("@BookId",book.BookId)

};

return DBhelp.Create().ExecuteNonQuery(sql, sp: sp);

}

//根据Id删除

public int DeleteBookInfo(string BookId)

{

string sql = @"delete from BorrowReturn where BookId=@BookId

delete from BookInfo where BookId=@BookId";

SqlParameter[] sp ={

new SqlParameter("@BookId",BookId)

};

return DBhelp.Create().ExecuteNonQuery(sql, sp: sp);

}