《数据库系统》课程设计报告

题目：图书销售管理系统

小组成员信息：

姓名	学号	班级	分工
宋渝杰	18340146	计科六班	负责实现“进货管理”模块
刘依澜	18340121	计科六班	负责实现“退货管理”模块
李梦圆	18340091	计科五班	负责实现“数据统计”模块
缪贝琪	18340131	计科六班	负责实现“销售管理”模块

一、开发环境与开发工具

win10 + Visual Studio 2019 C# + MySQL 8.0

二、系统需求分析

本次项目主要需要实现“图书销售管理系统”，需要记录图书的进货、退货、销售、统计功能。因此首先就是要建立“图书”表，基本信息如下：

表名：book

列名	数据类型	长度	是否允许为空	默认值	说明
bname	varchar	10	NO	NO	书名、主键
bprice	numeric	6, 1	NO	0	书本价格
bnum	int	4	NO	0	书本库存

之后由于图书的进货涉及到供货商及其对每本书的进货价格，因此接下来建立“供货商”表：

表名：supplier

列名	数据类型	长度	是否允许为空	默认值	说明
sname	varchar	10	NO	NO	供应商名、主键
bname	varchar	10	NO	NO	书名、主键、外码
sprice	numeric	6, 1	NO	0	进货价格

最后是由于进货、退货、销售等操作都需要生成对应的账单记录，统计功能也需要对所有的账单进行操作，因此建立“记录”表：

表名：record

列名	数据类型	长度	是否允许为空	默认值	说明
stime	varchar	25	NO	NO	记录时间、主键
bname	varchar	10	NO	NO	书名、外码
species	int	4	NO	NO	记录类型，0：销售，1：进货，2：退货
bprice	numeric	6,1	NO	0	当时的销售/进货/退货价格
bnum	int	4	NO	0	销售/进货/退货数量（退货数量为负数）

至此数据库部分建表需求已基本得到满足

三、功能需求分析

本次项目主要需要实现的 4 个功能如下：

- 进货：根据某种书籍的库存量及销售情况确定进货数量，根据供应商报价选择供应商。输出一份进货单并自动修改库存量，把本次进货的信息添加到进货库中
- 退货：顾客把已买的书籍退还给书店。输出一份退货单并自动修改库存量，把本次退货的信息添加到退货库中
- 销售：输入顾客要买书籍的信息，自动显示此书的库存量，如果可以销售，打印销售单并修改库存，同时把此次销售的有关信息添加到日销售库中
- 统计：根据销售情况输出统计的报表。一般内容为每月的销售总额、销售总量及排行榜

下文将逐步介绍这 4 个功能

进货：首先需要“选定一款书籍”，选定之后系统读取数据库，返回该书籍的库存量以及销售情况：

```
select bnum from book where bname = "bname"; # 库存量
select sum(bnum) from record where species != 1 and bname = "bname"; # 销售情况，包括销售和退货
```

之后根据供应商报价选择供应商，这里也需要系统读取供应商对这个书本的进货价：

```
select sname, sprice from supplier where bname = "bname"; # 读取供应商和进货价
```

最后是选定供应商和输入进货数量后，修改库存量并保存进货信息：

```
update book set bnum = bnum + "bnum" where bname = "bname"; # 修改库存量
insert into record values("stime", "bname", 1, "bprice", "bnum"); # 生成进货单
```

退货：首先需要“选定一款书籍”，选定之后系统读取数据库，返回该书籍的单价和库存量：

```
select bprice, bnum from book where bname = "bname"; # 读取书籍的单价和库存量
```

输入退货的数量后，确认退货，修改库存量并保存退货信息：

```
update book set bnum = bnum + "bnum" where bname = "bname"; # 修改库存量
insert into record values("stime", "bname", 2, "bprice", -"bnum"); # 生成退货单（注意退货数量为负数）
```

销售：首先需要“选定一款书籍”，选定之后系统读取数据库，返回该书籍的单价和库存量：

```
select bprice, bnum from book where bname = "bname"; # 读取书籍的单价和库存量
```

输入销售的数量后，确认销售，修改库存量并保存销售信息：

```
update book set bnum = bnum - "bnum" where bname = "bname"; # 修改库存量
insert into record values("stime", "bname", 0, "bprice", "bnum"); # 生成销售单
```

统计：统计部分我们实现的功能为“统计每月各个书籍的进货量、进货总额、销售（包括退货的影响）量、销售总额、净利润”，因此需要数据库对这些数据进行统计：

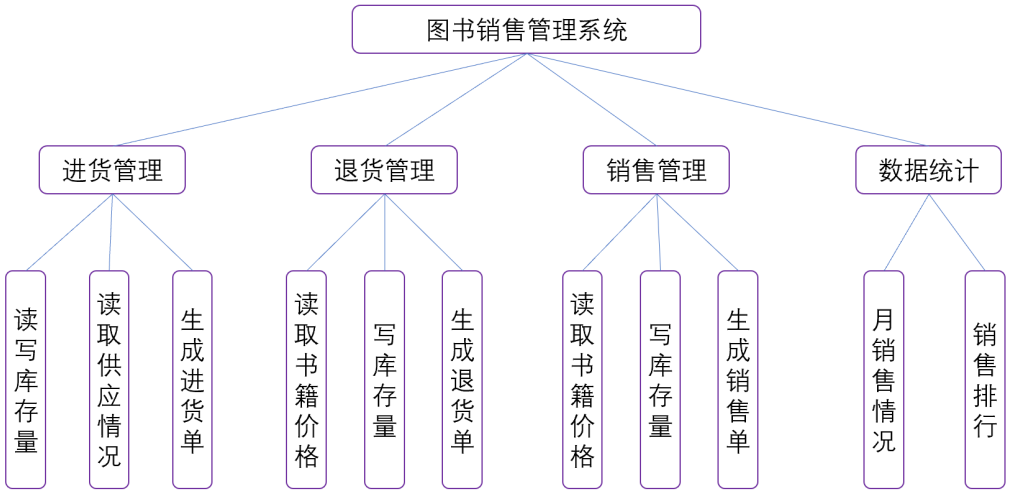
```
select bname, sum(bnum), sum(bprice*bnum) from record where species = 1 and
stime >= 'year-month-00-00-00-00' and stime <= 'year-month-31-23-59-59' group by
bname order by bname; # 统计各个书籍的进货量、进货总额
select bname, sum(bnum), sum(bprice*bnum) from record where species != 1 and
stime >= 'year-month-00-00-00-00' and stime <= 'year-month-31-23-59-59' group by
bname order by bname; # 统计各个书籍的销售（包括退货的影响）量、销售总额
```

净利润则计算上文统计出的销售总额 - 进货总额即可，交给 C# 客户端进行处理即可

```
for (i = 0; i < dt.Rows.Count; i++) {
    dt.Rows[i][5] = double.Parse(dt.Rows[i][4].ToString()) -
double.Parse(dt.Rows[i][2].ToString());
}
```

排行榜虽然可以通过 sql order 语句进行排序，但由于 C# dataGridView 的实用和优越性，可通过鼠标点击表格的列进行按列排序，因此排行榜部分也交给 C# 客户端处理

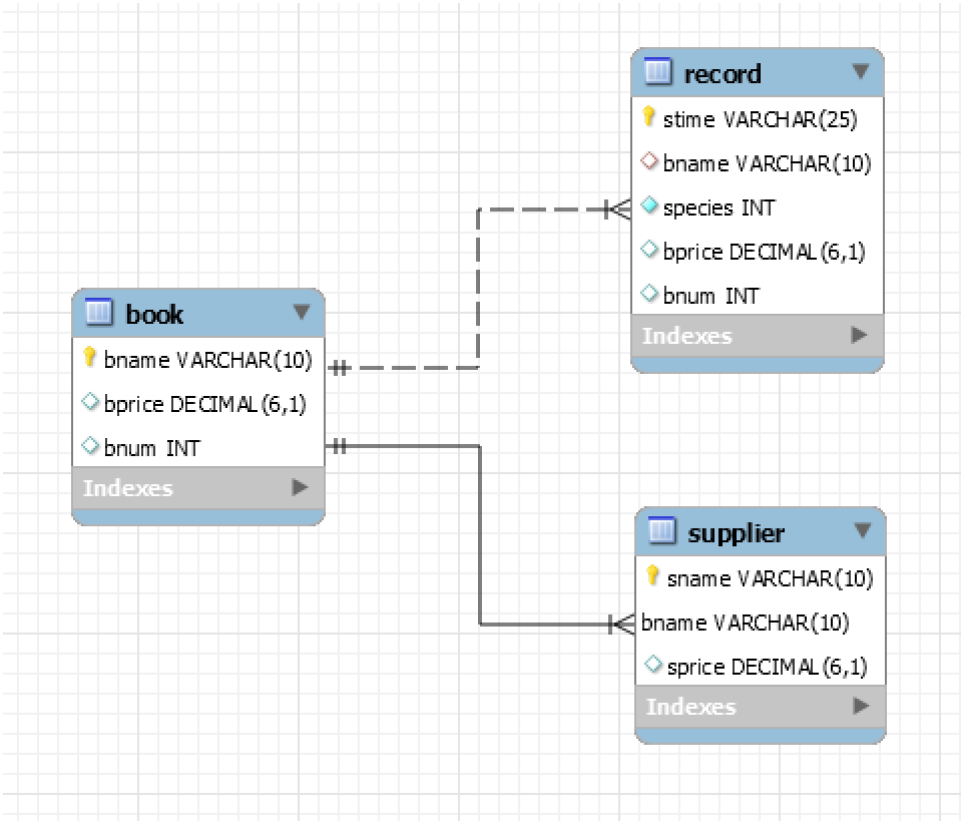
整个系统功能模块图如下：



四、系统设计

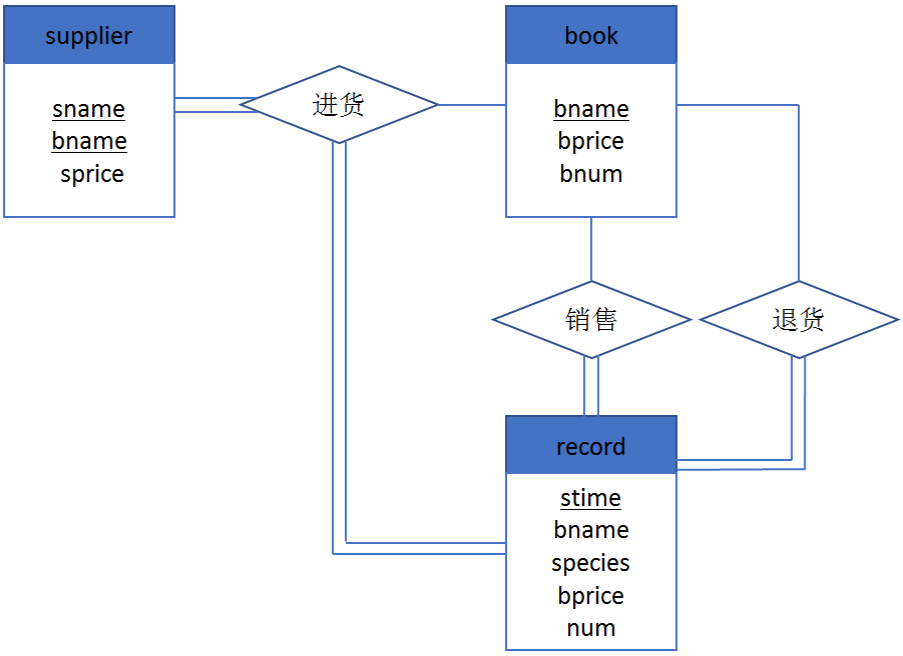
数据概念结构设计（系统 ER 图）

由 MySQL workbench 8.0 生成的系统 ER 图如下：



其中供应关系表 supplier 中的 bname、记录表 record 中的 bname 都是书本表 book 的外键

我们作出的 ER 图如下：



图书： book (bname, bprice, bnum)

供应商：supplier (sname, bname, sprice)

账单： record (stime, bname, species, bprice, num)

其中图书和供应商之间存在进货关系；而账单分别记录了图书的进货、销售和退货信息（分别对应 species 为 0、1、2 时），因此图书和账单之间存在进货、销售和退货三种关系。而供应商所供应给图书馆的图书必须是图书馆所需要的，因此 supplier 的 bname 是 book 的 bname 的外键；而账单记录的也必然是图书馆所拥有的书的相关记录，因此 record 的 bname 也是 book 的 bname 的外键。

数据库关系模式设计

首先回顾一下范式：

第一范式：表的每一列都是不可分割的原子数据项，而不能是集合，数组，记录等非原子数据项。可以看出我们的数据库表设计中，所有属性都不可再分，因此满足第一范式

第二范式：每一个非主属性完全函数依赖于任何一个候选码。可以看出我们的数据库表设计中，非主键部分完全依赖于主键（例如 supplier 中 sname 和 bname 确定后，sprice 才随之确定，且不依赖于这对候选码的子集），因此满足第二范式

第三范式：任何非主属性不依赖于其它非主属性。可以看出我们的数据库表设计中，非主键部分只依赖于主键（例如 book 中 bname 确定后，bprice 和 bnum 就随之确定，且这两个非主属性不存在互相依赖和传递依赖），因此满足第三范式

BCNF 范式：每个属性都不传递依赖于不包含它本身的候选键。由于我们的数据库表设计中，均只存在一个候选码，且满足第三范式（任何非主属性不依赖于其它非主属性），因此任何主属性也不依赖于不包含它本身的候选键，即满足 BCNF 范式

因此，我们设计的数据库表满足 BCNF 范式

数据库物理结构设计

物理存储引擎采用默认的 InnoDB 引擎，其支持事务，支持 MVCC 的行级锁，且无忌用场景。

而对于为表中字段选择合适的数据类型时，主要原则是：优先考虑数字类型，其次是日期或二进制类型，最后是字符类型。因此在书本价格、书本库存这些数据时，都采用数字类型进行存储；对于记录类型这些可以采取多种类型进行存储的数据，也采取了数字类型；对于书本名称、供应商名称等，只能采取字符类型；特别的是，对于记录 record 的记录时间这一数据，考虑到 C# 的读取当前时间的函数返回的基本都是字符串类型的数值，因此还是采用了字符类型进行存储，方便 C# 和 MySQL 的链接和数据传递。

五、系统功能的实现

主要功能模块的实现过程（简述）和运行界面

后台配置：运行程序时需要先在 MySQL 输入以下代码，新建数据库和输入测试数据

```
create database tsxs;
use tsxs;

create table book
(bname varchar(10),
bprice numeric(6,1) default 0,
bnum int default 0,
primary key(bname));

create table supplier
(sname varchar(10),
bname varchar(10),
sprice numeric(6,1) default 0,
primary key(sname,bname),
```

```
foreign key(bname) references book(bname));

create table record
(stime varchar(25),
bname varchar(10),
species int not null,
bprice numeric(6,1) default 0,
bnum int default 0,
primary key(stime),
foreign key(bname) references book(bname));

insert into book values('高等数学', 32.5, 100);
insert into book values('低等数学', 30, 80);
insert into book values('组合数学', 38, 120);
insert into book values('数学分析', 44.5, 110);
insert into book values('离散数学', 16.5, 190);
insert into supplier values('供应商A', '高等数学', 20);
insert into supplier values('供应商A', '低等数学', 22);
insert into supplier values('供应商A', '组合数学', 24);
insert into supplier values('供应商A', '数学分析', 26);
insert into supplier values('供应商A', '离散数学', 28);
insert into supplier values('供应商B', '高等数学', 18);
insert into supplier values('供应商B', '低等数学', 26);
insert into supplier values('供应商B', '组合数学', 22);
insert into supplier values('供应商B', '数学分析', 20);
insert into supplier values('供应商B', '离散数学', 14);
insert into supplier values('供应商C', '高等数学', 32);
insert into supplier values('供应商C', '低等数学', 28);
insert into supplier values('供应商C', '组合数学', 16);
insert into supplier values('供应商C', '数学分析', 25);
insert into supplier values('供应商C', '离散数学', 24.5);
insert into record values('2020-12-21-00-00-00', '高等数学', 1, 32.5, 10);
insert into record values('2020-12-21-00-00-01', '低等数学', 1, 30, 10);
insert into record values('2020-12-21-00-00-02', '组合数学', 1, 38, 10);
insert into record values('2020-12-21-00-00-03', '数学分析', 1, 44.5, 10);
insert into record values('2020-12-21-00-00-04', '离散数学', 1, 16.5, 10);
insert into record values('2020-12-21-00-00-05', '高等数学', 0, 32.5, 2);
insert into record values('2020-12-21-00-00-06', '低等数学', 0, 30, 4);
insert into record values('2020-12-21-00-00-07', '组合数学', 0, 38, 3);
insert into record values('2020-12-21-00-00-08', '数学分析', 0, 44.5, 2);
insert into record values('2020-12-21-00-00-09', '离散数学', 0, 16.5, 2);
```

登录：程序运行后进入登录页面，输入账号和密码后，系统将会以你输入的账号和密码的身份登录到主页面

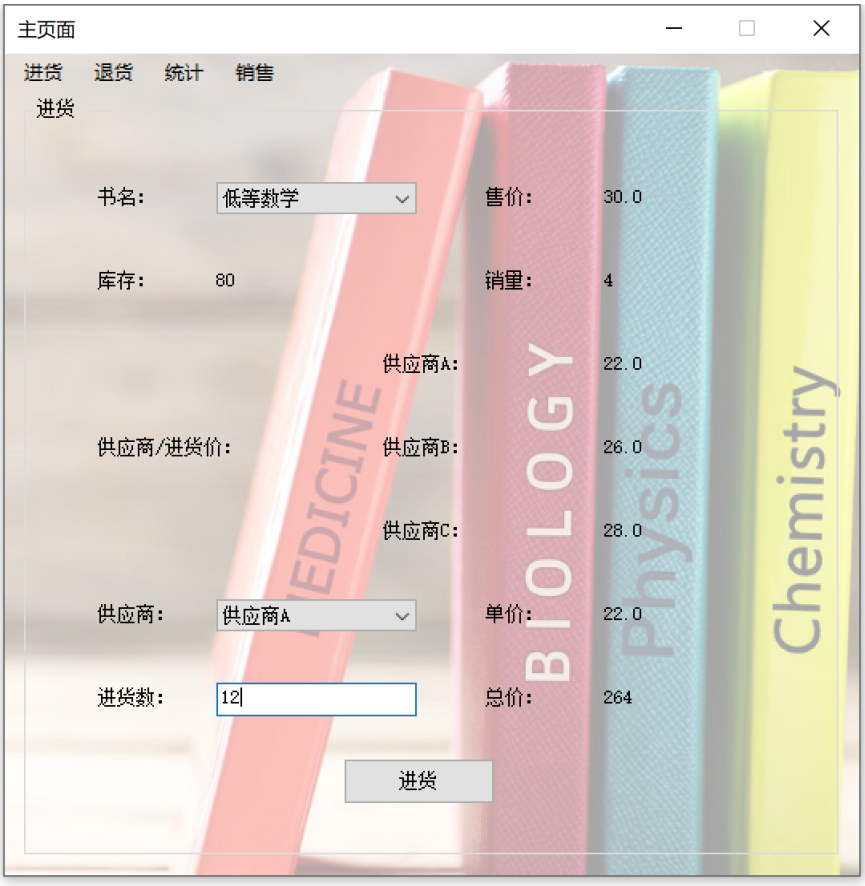
进货：菜单栏单击“进货”进入进货页面，在书名 comboBox 处选择对应的书籍后，程序使用 sql 语句读取数据库，读出该书籍的售价、库存和销量，以及供应商对于这本书籍的进货价格，然后将数值赋给对应的 label 标签。

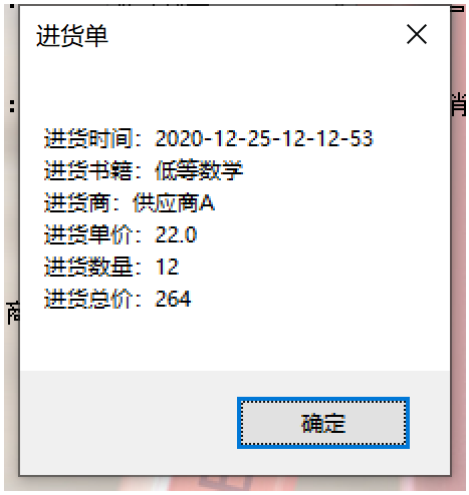
由于代码量过多，报告中仅放上读取书籍售价和库存的关键代码：

```
mySqlCommand = new MySqlCommand("select * from book where bname = '" +
comboBox2.Text + "';", conn);
mdr = mySqlCommand.ExecuteReader();
while (mdr.Read()) {
    label18.Text = mdr.GetString("bprice");
    label3.Text = mdr.GetString("bnum");
}
```

在供应商 comboBox 选择对应的供应商并输入进货数后，总价标签输出对应的进货价格，最后点击“进货”按钮，程序生成一个进货单，输出并将其保存进数据库，同时修改库存：

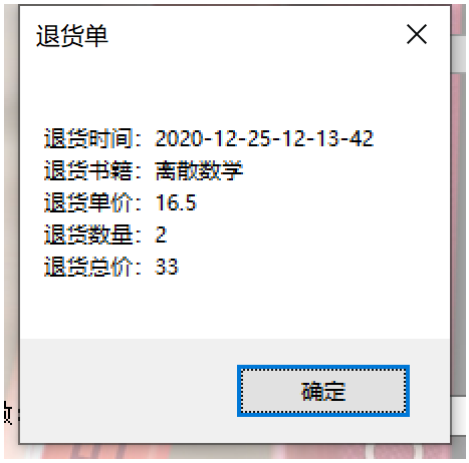
```
MySqlCommand mySqlCommand = new MySqlCommand("update book set bnum = bnum + " +
textBox2.Text + " where bname = '" + comboBox2.Text + "';", conn); // 修改库存
int result = mySqlCommand.ExecuteNonQuery();
string s = DateTime.Now.ToString("yyyy-MM-dd-hh-mm-ss");
mySqlCommand = new MySqlCommand("insert into record values('" + s + "', '" +
comboBox2.Text + "', 1, " + label14.Text + ", " + textBox2.Text + ");", conn);
// 进货单
result = mySqlCommand.ExecuteNonQuery();
```





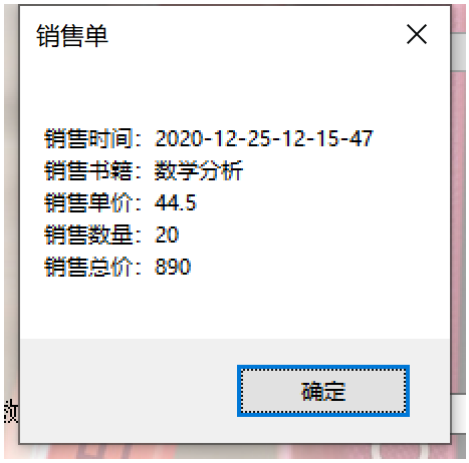
退货：书名处选定书籍后，程序显示库存和售价（即退货价格），接着输入退货数后，程序显示退货总价，最后点击退货，程序显示一个退货单，存入记录表并修改库存

```
MySqlCommand mySqlCommand = new MySqlCommand("select * from book where bname =
'" + comboBox3.Text + "';", conn); // 查询库存和售价
MySqlDataReader mdr = mySqlCommand.ExecuteReader();
mySqlCommand = new MySqlCommand("update book set bnum = bnum + " + textBox1.Text
+ " where bname = '" + comboBox3.Text + "';", conn); // 修改库存
int result = mySqlCommand.ExecuteNonQuery();
mySqlCommand = new MySqlCommand("insert into record values('" + s + "', '" +
comboBox3.Text + "', 2, " + label21.Text + ", -" + textBox1.Text + ");", conn);
// 退货单
result = mySqlCommand.ExecuteNonQuery();
```



销售：基本设计思路与流程和退货类似：书名处选定书籍后，程序显示库存和售价，接着输入销售数后，程序显示销售总价，最后点击销售，程序显示一个销售单，存入记录表并修改库存

```
MySqlCommand mySqlCommand = new MySqlCommand("select * from book where bname =
'" + comboBox4.Text + "';", conn); // 查询库存和售价
MySqlDataReader mdr = mySqlCommand.ExecuteReader();
mySqlCommand = new MySqlCommand("update book set bnum = bnum - " + textBox3.Text
+ " where bname = '" + comboBox4.Text + "';", conn); // 修改库存
int result = mySqlCommand.ExecuteNonQuery();
mySqlCommand = new MySqlCommand("insert into record values('" + s + "', '" +
comboBox4.Text + "', 0, " + label28.Text + ", " + textBox3.Text + ");", conn);
// 销售单
result = mySqlCommand.ExecuteNonQuery();
```



统计：选定统计的年月后，程序读取数据库相关数据，在中心表格处显示统计信息，包括各个书籍的书名、进货量、进货总额、销售（包括退货的影响）量、销售总额、净利润

这里体现出了为何退货时退货数量要设计为负数，在此处就可以直接求和计算出净利润

```
MySqlCommand mySqlCommand = new MySqlCommand("select bname, sum(bnum),
sum(bprice*bnum) from record where species = 1 and " + "stime >= '" +
dateTimePicker1.Value.Year + "-" + dateTimePicker1.Value.Month + "-00-00-00-00'
and " + "stime <= '" + dateTimePicker1.Value.Year + "-" +
dateTimePicker1.Value.Month + "-31-23-59-59' group by bname order by bname;",
conn); // 统计书名、进货量、进货总额
MySqlDataReader mdr = mySqlCommand.ExecuteReader();
mySqlCommand = new MySqlCommand("select bname, sum(bnum), sum(bprice*bnum) from
record where species != 1 and " + "stime >= '" + dateTimePicker1.Value.Year + "-
" + dateTimePicker1.Value.Month + "-00-00-00-00' and " + "stime <= '" +
dateTimePicker1.Value.Year + "-" + dateTimePicker1.Value.Month + "-31-23-59-59'
group by bname order by bname;", conn); // 统计书名、销售量、销售总额
mdr = mySqlCommand.ExecuteReader();
for (i = 0; i < dt.Rows.Count; i++) {
    dt.Rows[i][5] = double.Parse(dt.Rows[i][4].ToString()) -
double.Parse(dt.Rows[i][2].ToString()); // 统计每本书的净利润
}
```

六、总结

本次项目主要实现了“图书销售管理系统”，涉及到进货、退货、销售、统计功能，具体涉及的知识点如下：

- 数据库建表：主键、外键、非空、默认值，以及各种数据类型
- sql 语法：select、group、order、insert、update 等等
- ER 图、范式等一些分散的知识点

本次项目相比于其它的小实验来看，工程量大了不少，不仅是对数据库大部分知识点的融会贯通的应用，而且也涉及了一些额外的工作，比如使用 C# 实现图形界面，C# 和 MySQL 的链接通信等等。

整个实现过程花费了完整的几天时间，代码量也达到了四五百行，不仅需要学习 C# 新知识，而且需要复习数据库表的设计范式知识，以及利用此对最初设计的数据库表进行修改等等。