## 创建数据库BookShopping

CREATE DATABASE [BookShopping] ON PRIMARY

( NAME = N'BookShopping', FILENAME = N'C:\Program Files (x86)\Microsoft SQL Server\MSSQL10\_50.MSSQLSERVER\MSSQL\DATA\BookShopping.mdf' , SIZE = 3072KB , MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB )

LOG ON

( NAME = N'BookShopping\_log', FILENAME = N'C:\Program Files (x86)\Microsoft SQL Server\MSSQL10\_50.MSSQLSERVER\MSSQL\DATA\BookShopping\_log.ldf' , SIZE = 1024KB , MAXSIZE = 2048GB , FILEGROWTH = 10%)

GO

## 创建表

### 管理员信息表AdminInfo

CREATE TABLE [dbo].[AdminInfo](

[adminId] [int] IDENTITY(1,1) NOT NULL,

[adminName] [nvarchar](20) NULL,

[passWord] [int] NULL,

CONSTRAINT [PK\_Admin] PRIMARY KEY CLUSTERED

(

[adminId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

### 图书信息表BookInfo

CREATE TABLE [dbo].[BookInfo](

[id] [int] IDENTITY(1,1) NOT NULL,

[bookName] [nvarchar](50) NULL,

[bookPrice] [money] NULL,

[category] [nvarchar](300) NULL,

[mark] [nvarchar](300) NULL,

[writer] [nvarchar](300) NULL,

[storeMount] [int] NULL,

CONSTRAINT [PK\_BookInfo] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

### 购书订单表BookOrder

CREATE TABLE [dbo].[BookOrder](

[bookOrderId] [int] IDENTITY(1,1) NOT NULL,

[customerId] [int] NOT NULL,

[myBookId] [int] NOT NULL,

[customerName] [nvarchar](50) NULL,

[customerAddress] [nvarchar](300) NULL,

[customerTelephone] [nvarchar](20) NULL,

[notic] [nvarchar](200) NULL,

[subDate] [date] NULL,

[logistics] [nvarchar](20) NULL,

[paymentStatus] [nvarchar](20) NULL,

CONSTRAINT [PK\_BookOrder] PRIMARY KEY CLUSTERED

(

[bookOrderId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[BookOrder] WITH CHECK ADD CONSTRAINT [FK\_BookOrder\_BookInfo] FOREIGN KEY([customerId])

REFERENCES [dbo].[UserInfo] ([userId])

GO

ALTER TABLE [dbo].[BookOrder] CHECK CONSTRAINT [FK\_BookOrder\_BookInfo]

GO

ALTER TABLE [dbo].[BookOrder] WITH CHECK ADD CONSTRAINT [FK\_BookOrder\_BookInfo1] FOREIGN KEY([myBookId])

REFERENCES [dbo].[BookInfo] ([id])

GO

ALTER TABLE [dbo].[BookOrder] CHECK CONSTRAINT [FK\_BookOrder\_BookInfo1]

GO

ALTER TABLE [dbo].[BookOrder] ADD CONSTRAINT [DF\_BookOrder\_Pocessing] DEFAULT ('未发货') FOR [logistics]

GO

ALTER TABLE [dbo].[BookOrder] ADD CONSTRAINT [DF\_BookOrder\_payment status] DEFAULT ('未支付') FOR [paymentStatus]

GO

### 购物车Cart

CREATE TABLE [dbo].[Cart](

[bookId] [int] NOT NULL,

[number] [int] NOT NULL,

[totalPrice] [money] NOT NULL,

CONSTRAINT [PK\_Cart\_1] PRIMARY KEY CLUSTERED

(

[bookId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[Cart] WITH CHECK ADD CONSTRAINT [FK\_Cart\_BookInfo] FOREIGN KEY([bookId])

REFERENCES [dbo].[BookInfo] ([id])

GO

ALTER TABLE [dbo].[Cart] CHECK CONSTRAINT [FK\_Cart\_BookInfo]

GO

### 用户信息表UserInfo

CREATE TABLE [dbo].[UserInfo](

[userId] [int] IDENTITY(1,1) NOT NULL,

[userName] [nvarchar](50) NULL,

[passWord] [nvarchar](20) NULL,

[bak] [nvarchar](100) NULL,

CONSTRAINT [PK\_UserInfo] PRIMARY KEY CLUSTERED

(

[userId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

## 创建约束

USE [BookShopping]

GO

ALTER TABLE [dbo].[BookOrder] ADD CONSTRAINT [DF\_BookOrder\_payment status] DEFAULT ('未支付') FOR [paymentStatus]

GO

ALTER TABLE [dbo].[BookOrder] ADD CONSTRAINT [DF\_BookOrder\_Pocessing] DEFAULT ('未发货') FOR [logistics]

GO

## 添加数据

（1）

INSERT INTO [BookShopping].[dbo].[BookInfo]

([bookName]

,[bookPrice]

,[category]

,[mark]

,[writer]

,[storeMount])

VALUES

(<bookName, nvarchar(50),>

,<bookPrice, money,>

,<category, nvarchar(300),>

,<mark, nvarchar(300),>

,<writer, nvarchar(300),>

,<storeMount, int,>)

GO

（2）

INSERT INTO [BookShopping].[dbo].[BookOrder]

([customerId]

,[myBookId]

,[customerName]

,[customerAddress]

,[customerTelephone]

,[notic]

,[subDate]

,[logistics]

,[paymentStatus])

VALUES

(<customerId, int,>

,<myBookId, int,>

,<customerName, nvarchar(50),>

,<customerAddress, nvarchar(300),>

,<customerTelephone, nvarchar(20),>

,<notic, nvarchar(200),>

,<subDate, date,>

,<logistics, nvarchar(20),>

,<paymentStatus, nvarchar(20),>)

GO

（3）

INSERT INTO [BookShopping].[dbo].[Cart]

([bookId]

,[number]

,[totalPrice])

VALUES

(<bookId, int,>

,<number, int,>

,<totalPrice, money,>)

GO

（4）

INSERT INTO [BookShopping].[dbo].[UserInfo]

([userName]

,[passWord]

,[bak])

VALUES

(<userName, nvarchar(50),>

,<passWord, nvarchar(20),>

,<bak, nvarchar(100),>)

GO

## 创建触发器

（1）

CREATE TRIGGER [dbo].[trig1]

ON [dbo].[BookInfo]

for update

AS

if(update(storeMount))

BEGIN

declare @mount int

set @mount=(select storeMount from inserted)

if(@mount<0)

begin

print'库存不足！'

rollback

END

End

（2）

CREATE TRIGGER [dbo].[tri\_1]

ON [dbo].[Cart]

FOR INSERT,UPDATE

AS

BEGIN

declare @mount int,@price int,@total int

set @mount=(select number from inserted)

set @price=(select bookPrice from BookInfo,inserted where BookInfo.id=inserted.bookId)

set @total=@mount\*@price

update cart set totalPrice=@total where bookId=(select bookId from inserted)

END

## 创建存储过程

（1）

CREATE PROCEDURE [dbo].[bookcheck]

@name nvarchar(20)

AS

BEGIN

SELECT \* from BookInfo

where bookName like '%'+@name+'%';

END

（2）

CREATE PROCEDURE [dbo].[customercheck]

@name nvarchar(50),

@customerName nvarchar(50) output,

@customerAddress nvarchar(300) output,

@customerTelephone nvarchar(20) output,

@notic nvarchar(200) output

AS

begin

SELECT

@customerName=customerName,@customerAddress=customerAddress,

@customerTelephone=customerTelephone,@notic=notic

from BookOrder

where BookOrder.customerName=@name

end

（3）

create PROCEDURE [dbo].[ordercheck]

@name nvarchar(50),

@subDate date output,

@bookName nvarchar(50) output,

@bookPrice money output,

@totalPrice money output,

@logistice nvarchar(20) output,

@paymentStatus nvarchar(20) output

AS

begin

SELECT

@subDate=subDate,@bookName=bookName,@bookPrice=bookPrice,@totalPrice=totalPrice,@logistice=logistics,@paymentStatus=paymentStatus

from BookInfo,BookOrder,Cart

where BookInfo.id=BookOrder.myBookId and Cart.bookId=BookInfo.id and Cart.bookId=BookOrder.myBookId and BookOrder.customerName=@name

end

（4）

CREATE PROCEDURE [dbo].[pay]

@cost money,

@bookOrderId int,

@logistics nvarchar(20) output,

@paymentStatus nvarchar(20) output,

@totalPrice money output

AS

BEGIN

update BookOrder

set paymentStatus='已付款',logistics='发货中' where bookOrderId=@bookOrderId if @cost>=@totalPrice

SELECT @paymentStatus=paymentStatus,@logistics=logistics,@totalPrice=totalPrice from BookOrder,Cart

where Cart.totalPrice=@totalPrice

and BookOrder.bookOrderId=@bookOrderId

END

## 创建用户

（1）创建shopkeeper

CREATE LOGIN [shopkeeper] WITH PASSWORD=, DEFAULT\_DATABASE=[master], DEFAULT\_LANGUAGE=[简体中文], CHECK\_EXPIRATION=ON, CHECK\_POLICY=ON

GO

ALTER LOGIN [shopkeeper] DISABLE

GO

CREATE USER [shopkeeper] FOR LOGIN [shopkeeper] WITH DEFAULT\_SCHEMA=[dbo]

GO

（2）创建customer1

CREATE LOGIN [customer1] WITH PASSWORD=, DEFAULT\_DATABASE=[master], DEFAULT\_LANGUAGE=[简体中文], CHECK\_EXPIRATION=ON, CHECK\_POLICY=ON

GO

ALTER LOGIN [customer1] DISABLE

GO

CREATE USER [customer1] FOR LOGIN [customer1] WITH DEFAULT\_SCHEMA=[dbo]

GO

（3）创建customer2

CREATE LOGIN [customer2] WITH PASSWORD=, DEFAULT\_DATABASE=[master], DEFAULT\_LANGUAGE=[简体中文], CHECK\_EXPIRATION=ON, CHECK\_POLICY=ON

GO

ALTER LOGIN [customer2] DISABLE

GOCREATE USER [customer2] FOR LOGIN [customer2] WITH DEFAULT\_SCHEMA=[dbo]

## 设置权限

（1）

use BookShopping

grant select,insert,delete,update on BookOrder to customer1

grant select,insert,delete on Cart to customer1

grant exec to customer1

（2）

use BookShopping

grant select,insert,delete,update on BookOrder to customer2

grant select,insert,delete on Cart to customer2

grant exec to customer2

（3）

use BookShopping

grant db\_owner to shopkeeper