

数据库系统开发实验

第七章实验、数据库编程

- 班级：07111606
- 学号：1120161881
- 实验人：张启洋
- 实验日期：2018/09/26 - 2018/09/28

目录

数据库系统开发实验

第七章实验、数据库编程

目录

实验二、订单管理

- 一、实验方法
 - (一) 开发环境
 - (二) 问题回答
 - (三) 添加代码及说明

二、实验结果

三、实验总结

2018/09/28 23:14 实验人：张启洋 学号：1120161881

实验三、购物车

- 一、实验方法
 - (一) 开发环境
 - (二) 问题回答
 - (三) 添加代码及说明

二、实验结果

三、实验总结

2018/09/29 19:18 实验人：张启洋 学号：1120161881

实验二、订单管理

一、实验方法

(一) 开发环境

- 操作系统: Windows 10 Home China
- 服务器和实例名称: DWQETHINKPAD
- 数据库服务器: Microsoft SQL Server 2017 Developer Edition
- 数据库客户端: SQL Server Management Studio v17.8.1
- 编程工具: Microsoft Visual Studio Community 2017 Version 15.8.2
- 编程语言: C# (.NET Framework 4.6.1)

(二) 问题回答

1. 使用哪种数据提供程序?

SQL Server数据提供程序 (System.Data.SqlClient)

2. 使用的数据连接对象是哪一个? 连接对象是如何建立的? 最后生成的连接对象中的连接字符串是什么? 代表什么含义?

- 数据连接对象: sqlConnection
- 建立方法: 创建数据适配器是自动建立
- 连接字符串:

```
1 Data Source=DwqeThinkPad;Initial Catalog=AdventureWorks2016;Integrated Security=True
```

- 字符串含义: 连接服务器是DwqeThinkPad默认实例
数据库是AdventureWorks2016
使用Windows认证登录

3. 使用的数据适配器对象是什么? 其中的查询或更新语句是什么? 如果有参数, 则参数是如何处理的?

1) 数据适配器: headerDataAdapter

- 查询语句:

```
1 SELECT Sales.SalesOrderHeader.*  
2 FROM Sales.SalesOrderHeader
```

- 更新、插入、删除语句: 由Visual Studio自动生成
- 参数也由Visual Studio自动生成

2) 数据适配器: detailDataAdapter

- 查询语句

```
1  SELECT    Sales.SalesOrderDetail.*
2  FROM      Sales.SalesOrderDetail
```

- 更新、插入、删除语句: 由Visual Studio自动生成
- 参数也由Visual Studio自动生成

4. 使用的数据集对象是什么? 数据集有哪些数据表? 数据表是由哪些适配器对象生成的? (或采用其它方法)

- 数据集对象: dataset
- 包含数据表: SalesOrderHeader (适配器: headerDataAdapter) , SalesOrderDetail (适配器: detailDataAdapter)

(三) 添加代码及说明

1. 加载数据集

```
1  private void Form1_Load(object sender, EventArgs e)
2  {
3      //调用两个数据适配器填充数据集
4      headerDataAdapter.Fill(dataset, "SalesOrderHeader");
5      detailDataAdapter.Fill(dataset, "SalesOrderDetail");
6      //调用数据绑定, 在窗口控件中加载数据
7      detailBindingSource.DataSource = headerBindingSource;
8      detailBindingSource.DataMember = "SalesOrderHeader_SalesOrderDetail";
9  }
```

2. 删除按钮

```
1  private void btnDelete_Click(object sender, EventArgs e)
2  {
3      //当订单细节不为空, 读取当前选中行索引, 不是最后的空行, 就执行删除操作
4      if (this.dgvDetail.SelectedRows.Count > 0 && this.dgvDetail.SelectedRows[0].Index !=
5          this.dgvDetail.Rows.Count - 1)
6      {
7          //使用系统函数, 删除选中行数据
8          this.dgvDetail.Rows.RemoveAt(this.dgvDetail.SelectedRows[0].Index);
9      }
```

3. 保存按钮

```

1 private void btnSave_Click(object sender, EventArgs e)
2 {
3     //数据集未被修改, 直接结束
4     if (!dataset.HasChanges())
5         return;
6     //如果数据集被修改, 执行修改操作
7     try
8     {
9         //遍历每一行数据, 如果被数据修改, 修改数据库中数据
10        foreach (DataRow dataRow in this.dataset.SalesOrderDetail.Rows)
11            if (dataRow.RowState == DataRowState.Added)
12                if (dataRow["rowguid"].Equals(System.DBNull.Value))
13                    dataRow["rowguid"] = Guid.NewGuid();
14        //调用数据适配器更新数据集
15        headerDataAdapter.Update(dataset);
16        detailDataAdapter.Update(dataset);
17        MessageBox.Show("保存成功! ", "保存");
18    }
19    //抓取错误信息, 失败返回错误信息
20    catch (Exception ex)
21    {
22        MessageBox.Show("保存失败! " + ex.Message, "保存");
23    }
24 }

```

4. 关闭按钮

```

1 private void btnClose_Click(object sender, EventArgs e)
2 {
3     //数据集未被修改, 退出当前程序
4     if (!dataset.HasChanges())
5         Application.Exit();
6     //弹出对话框, 用户选择是否保存
7     DialogResult result = MessageBox.Show(
8         "此次修改保存, 是否保存将最新修改保存至数据库? ", "未保存",
9         MessageBoxButtons.YesNo,
10        MessageBoxIcon.Warning
11    );
12    //如果选择确认按钮, 保存当前程序并退出
13    if (result == DialogResult.Yes)
14        btnSave_Click(sender, e);
15    Application.Exit();
16 }

```

二、实验结果

1. 订单明细查询: 点击上方订单其中一行数据, 在下方订单明细中查看到详细信息

订单管理

订单

	SalesOrderID	RevisionNumber	OrderDate	DueDate	ShipDate	Status	OnlineOrderFlag	SalesOrderNumber	PurchaseOrderNum
▶	43659	10	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043659	P0522145787
	43660	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043660	P018850127500
	43661	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043661	P018473189620
	43662	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043662	P018444174044
	43663	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043663	P018009186470

订单明细

	SalesOrderID	SalesOrderDetail	CarrierTracking	OrderQty	ProductID	SpecialOfferID	UnitPrice	UnitPriceDiscow	LineTotal
▶	43659	1	4911-403C-98	1	776	1	1000.0000	0.0000	1000.000000
	43659	3	4911-403C-98	1	778	1	2024.9940	0.0000	2024.994000
	43659	4	4911-403C-98	1	771	1	2039.9940	0.0000	2039.994000
	43659	5	4911-403C-98	1	772	1	2039.9940	0.0000	2039.994000
	43659	6	4911-403C-98	2	773	1	2039.9940	0.0000	4079.968000

删除

保存

关闭

2. 订单明细插入：点击下方订单明细中最后一行，严格按照数据格式逐列，输入数据，其中LineTotal和rowguid可以为空值，完成输入后点击保存按钮

订单管理

订单

	SalesOrderID	RevisionNumber	OrderDate	DueDate	ShipDate	Status	OnlineOrderFlag	SalesOrderNumber	PurchaseOrderNum
▶	43659	10	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043659	P0522145787
	43660	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043660	P018850127500
	43661	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043661	P018473189620
	43662	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043662	P018444174044
	43663	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043663	P018009186470

订单明细

	CarrierTracking	OrderQty	ProductID	SpecialOfferID	UnitPrice	UnitPriceDiscow	LineTotal	rowguid	ModifiedDate
	4911-403C-98	6	709	1	5.7000	0.0000	34.200000	ac769034-3c2...	2011/5/31
	4911-403C-98	2	712	1	5.1865	0.0000	10.373000	06a66921-6b9...	2011/5/31
	4911-403C-98	4	711	1	20.1865	0.0000	80.746000	0e371ee3-253...	2011/5/31
▶	4911-403C-98	5	677	1	2.0000	0.0000			2018/9/28
*									

删除

保存

关闭

如果成功保存，弹出提示窗口

保存

×

保存成功!

确定

3. 订单明细更新：点击下方订单明细中需要修改的数据，输入修改后数据，完成输入后点击保存按钮，此时会触发数据库中的触发器，例如UnitPrice大于ListPrice时，就会触发第四章实验中创建的触发器，如果没有触发器被触发，则保存成功

订单管理

订单

	SalesOrderID	RevisionNumber	OrderDate	DueDate	ShipDate	Status	OnlineOrderFlag	SalesOrderNumber	PurchaseOrderNumber
▶	43659	11	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043659	P0622145787
	43660	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043660	P018850127500
	43661	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043661	P018473189620
	43662	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043662	P018444174044
	43663	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043663	P018009186470

订单明细

	SalesOrderID	SalesOrderDetailID	CarrierTrackingID	OrderQty	ProductID	SpecialOfferID	UnitPrice	UnitPriceDiscount	LineTotal
▶	43659	1	4911-403C-98	1	776	1	50000	0.0000	2039.994000
	43659	4	4911-403C-98	1	771	1	2039.9940	0.0000	2039.994000
	43659	5	4911-403C-98	1	772	1	2039.9940	0.0000	2039.994000
	43659	6	4911-403C-98	2	773	1	2039.9940	0.0000	4079.988000
	43659	7	4911-403C-98	1	774	1	2039.9940	0.0000	2039.994000

保存失败! 事务在触发器中结束。批处理已中止。

确定

删除 保存 关闭

4. 订单明细删除：点击下方订单明细中需要删除的一行数据，点击删除按钮，再点击保存按钮，第一行数据成功被删除

订单管理

订单

	SalesOrderID	RevisionNumber	OrderDate	DueDate	ShipDate	Status	OnlineOrderFlag	SalesOrderNumber	PurchaseOrderNumber
▶	43659	11	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043659	P0622145787
	43660	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043660	P018850127500
	43661	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043661	P018473189620
	43662	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043662	P018444174044
	43663	8	2011/5/31	2011/6/12	2011/6/7	5	<input type="checkbox"/>	S043663	P018009186470

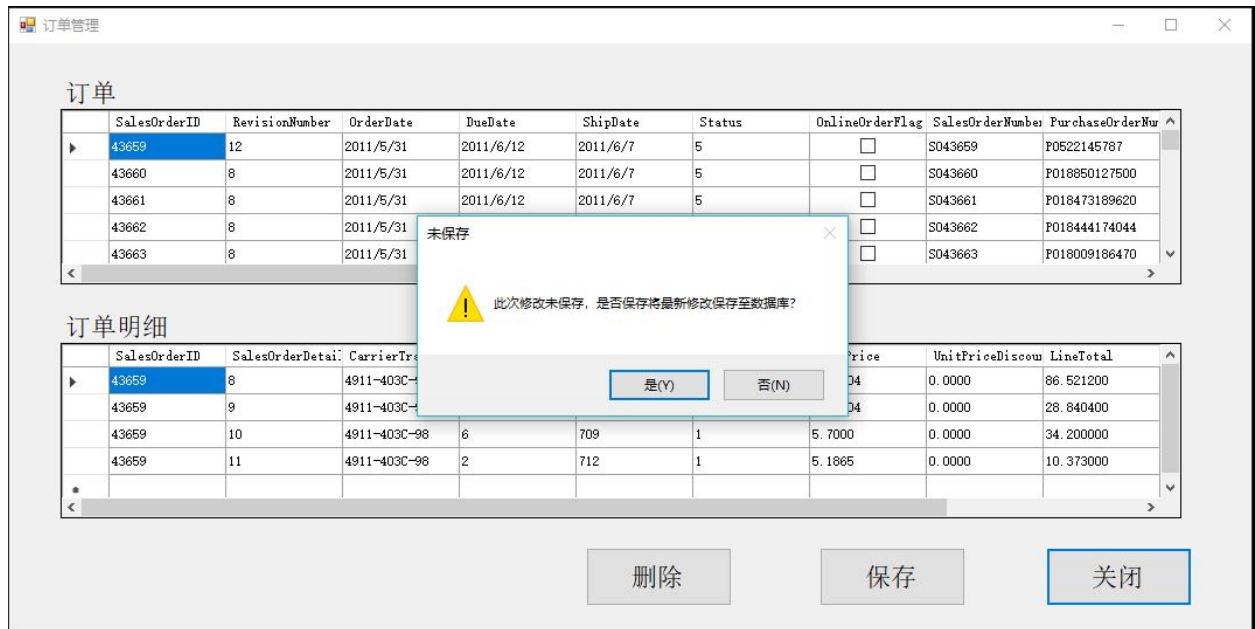
订单明细

	SalesOrderID	SalesOrderDetailID	CarrierTrackingID	OrderQty	ProductID	SpecialOfferID	UnitPrice	UnitPriceDiscount	LineTotal
▶	43659	4	4911-403C-98	1	771	1	2039.9940	0.0000	2039.994000
	43659	5	4911-403C-98	1	772	1	2039.9940	0.0000	2039.994000
	43659	6	4911-403C-98	2	773	1	2039.9940	0.0000	4079.988000
	43659	7	4911-403C-98	1	774	1	2039.9940	0.0000	2039.994000
	43659	8	4911-403C-98	3	714	1	28.8404	0.0000	86.521200

删除 保存 关闭

5. 保存按钮：在对订单明细修改后，需要点击保存按钮，修改数据库，否则在关闭后再次打开时，发现修改未成功，这是因为在点击保存按钮前，只是修改了数据集，点击保存按钮后，才会修改数据库，下次打开时发现成功修改数据

6. 关闭按钮：如果对数据集做出修改后没有点击保存按钮，在点击关闭按钮时，会弹出对话框，提示用户进行保存，点击按钮是，保存并退出程序，显示保存结果，点击按钮否，直接退出程序



三、实验总结

- 在配置数据库适配器时，发现界面中的数据集未被填充，在修改无果后，重新做了一遍，发现是调用的函数名被意外修改，但编译器不会报错，所以问题较难发现
- 可以改进的地方就是应该在点击上方关闭窗口按钮时，已设置与下方关闭按钮相同的功能，更加方便用户使用；在窗口最大化后，里面控件未发生变化，可以将空间设置为自适应窗口大小，更加美观

2018/09/28 23:14

实验人：张启洋

学号：1120161881

实验三、购物车

一、实验方法

(一) 开发环境

- 操作系统：Windows 10 Home China
- 服务器和实例名称：DWQETHINKPAD
- 数据库服务器：Microsoft SQL Server 2017 Developer Edition
- 数据库客户端：SQL Server Management Studio v17.8.1
- 编程工具：Microsoft Visual Studio Community 2017 Version 15.8.2
- 编程语言：C# (.NET Framework 4.6.1)

(二) 问题回答

1. 使用哪种数据提供程序？

SQL Server数据提供程序 (System.Data.SqlClient)

2. 使用的数据连接对象是哪一个？连接对象是如何建立的？最后生成的连接对象中的连接字符串是什么？代表什么含义？

- 数据连接对象：sqlConnection
- 建立方法：创建数据适配器是自动建立
- 连接字符串：

```
1 Data Source=DwqeThinkPad;Initial Catalog=AdventureWorks2016;Integrated Security=True
```

- 字符串含义：连接服务器是DwqeThinkPad默认实例
数据库是AdventureWorks2016
使用Windows认证登录

3. 使用的数据适配器对象是什么？其中的查询或更新语句是什么？如果有参数，则参数是如何处理的？

1) 数据适配器：ShoppingDataAdapter

- 查询语句：

```
1 SELECT Sales.ShoppingCartItem.*  
2 FROM Sales.ShoppingCartItem
```

- 更新、插入、删除语句：由Visual Studio自动生成
- 参数也由Visual Studio自动生成

2) 数据适配器: ProductionDataAdapter

- 查询语句

```
1 SELECT Production.ProductInventory.*
2 FROM Production.ProductInventory
```

- 更新、插入、删除语句: 由Visual Studio自动生成
- 参数也由Visual Studio自动生成

4. 使用的数据集对象是什么? 数据集中有哪些数据表? 数据表是由哪些适配器对象生成的? (或采用其它方法)

- 数据集对象: dataset
- 包含数据表: ShoppingCartItem (适配器: ShoppingDataAdapter) , ProductInventory (适配器: ProductionDataAdapter)

(三) 添加代码及说明

1. 创建添加商品存储过程

```
1 USE [AdventureWorks2016]
2 GO
3 --删除已有此存储过程
4 IF OBJECT_ID('[AddProduction]', 'P') IS NOT NULL
5     DROP PROCEDURE [AddProduction]
6 GO
7 --重新创建存储过程
8 CREATE PROCEDURE [AddProduction]
9     @ProductID int,
10    @LocationID int,
11    @Quantity int
12
13 AS
14     DECLARE @TotalPrcCNT int --添加物品总量
15
16     SELECT @TotalPrcCNT = P.Quantity
17     FROM [Production].[ProductInventory] P
18     WHERE P.ProductID = @ProductID AND P.LocationID = @LocationID
19     --判定添加物品数量是否大于库存数量
20     IF @TotalPrcCNT < @Quantity
21         RETURN -1
22
23     DECLARE @ShoppingCartID nvarchar(50) = CONVERT(nvarchar(50), @LocationID)
24
25     --如果购物车已有该物品, 更新数量, 没有该物品, 添加新行
26     IF EXISTS (
27         SELECT * FROM [Sales].[ShoppingCartItem]
28         WHERE [ProductID] = @ProductID AND [ShoppingCartID] = @ShoppingCartID
```

```

29     )
30     UPDATE [Sales].[ShoppingCartItem]
31     SET [Quantity] += @Quantity
32     WHERE [ProductID] = @ProductID
33 ELSE
34     INSERT INTO [Sales].[ShoppingCartItem] (
35         ShoppingCartID, Quantity, ProductID, DateCreated, ModifiedDate
36     )
37     VALUES (
38         @ShoppingCartID, @Quantity, @ProductID, GETDATE(), GETDATE()
39     )
40
41     SET @TotalPrcCNT = @Quantity
42     --更新仓库中数据
43     UPDATE [Production].[ProductInventory]
44     SET [Quantity] -= @TotalPrcCNT
45     WHERE [Production].[ProductInventory].[ProductID] = @ProductID AND
46         [Production].[ProductInventory].[LocationID] = @LocationID
47
48     RETURN 1
49
50 GO

```

2. 创建删除商品存储过程

```

1  --删除已有此存储过程
2  IF OBJECT_ID('[DeleteProduction]', 'P') IS NOT NULL
3      DROP PROCEDURE DeleteProduction
4  GO
5  --重新创建存储过程
6  CREATE PROCEDURE DeleteProduction
7      @ShoppingCartItemID int,
8      @ShoppingCarID int,
9      @Quantity int,
10     @ProductID int
11  AS
12     --更新仓库中数据
13     UPDATE [Production].[ProductInventory]
14     SET Quantity += @Quantity
15     WHERE [Production].[ProductInventory].[ProductID] = @ProductID AND
16         [Production].[ProductInventory].[LocationID] = Convert(smallint, @ShoppingCarID)
17
18     --删除购物车中记录
19     DELETE FROM [Sales].[ShoppingCartItem]
20     WHERE [Sales].[ShoppingCartItem].[ShoppingCartItemID] = @ShoppingCartItemID
21 GO

```

3. 加载数据集

```

1 private void Form1_Load(object sender, EventArgs e)
2 {
3     ProductionDataAdapter.Fill(dataset, "ProductInventory");
4     ShoppingDataAdapter.Fill(dataset, "ShoppingCartItem");
5 }

```

4. 添加按钮

```

1 private void btnAdd_Click(object sender, EventArgs e)
2 {
3     //读取数据
4     int ProductID = Int32.Parse(dgvProduction.CurrentRow.Cells[0].Value.ToString());
5     int LocationID = Int32.Parse(dgvProduction.CurrentRow.Cells[1].Value.ToString());
6     int Quantity = Int32.Parse(ProNum.Text);
7     //判定是否输入
8     if (ProNum.Text.Trim() == String.Empty)
9     {
10         MessageBox.Show("请输入要加入购物车的商品数量!", "提示",
11             MessageBoxButtons.OK, MessageBoxIcon.Warning);
12     }
13     return;
14 }
15 //添加调用存储过程命令
16 SqlCommand cmd_AddProduction = new SqlCommand("AddProduction", this.sqlConnection);
17 cmd_AddProduction.CommandType = CommandType.StoredProcedure;
18 //传输数据
19 cmd_AddProduction.Parameters.AddWithValue("@ProductID", SqlDbType.Int);
20 cmd_AddProduction.Parameters["@ProductID"].Value = ProductID;
21 cmd_AddProduction.Parameters.AddWithValue("@LocationID", SqlDbType.Int);
22 cmd_AddProduction.Parameters["@LocationID"].Value = LocationID;
23 cmd_AddProduction.Parameters.AddWithValue("@Quantity", SqlDbType.Int);
24 cmd_AddProduction.Parameters["@Quantity"].Value = Quantity;
25 //打开数据连接
26 this.sqlConnection.Open();
27 int return_value = cmd_AddProduction.ExecuteNonQuery(); //读取存储过程返回值
28 this.sqlConnection.Close();
29
30 //根据返回值提示信息
31 if (return_value == -1)
32 {
33     MessageBox.Show("加入购物车失败! 加入购物车数量不得超过库存数量", "提示",
34         MessageBoxButtons.OK, MessageBoxIcon.Information);
35 }
36 return;
37 }
38 else
39 {
40     MessageBox.Show("您已将商品添加加入购物车!", "提示",
41         MessageBoxButtons.OK, MessageBoxIcon.Information);
42 }

```

```

43         this.ProductionDataAdapter.Fill(dataset);
44         this.ShoppingDataAdapter.Fill(dataset);
45         return;
46     }
47 }

```

5. 删除按钮

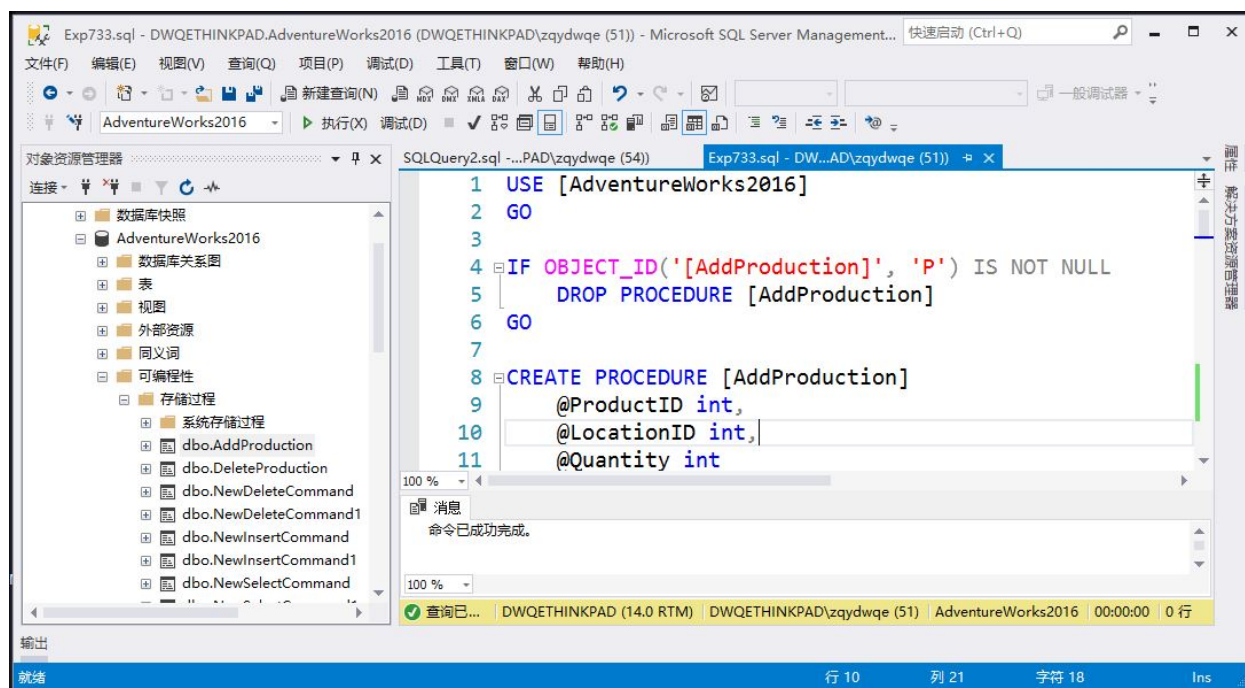
```

1 private void btnDelete_Click(object sender, EventArgs e)
2 {
3     //读取数据
4     int ShoppingCartItemID =
5     Int32.Parse(dgvShopping.CurrentRow.Cells[0].Value.ToString());
6     int ShoppingCartID = Int32.Parse(dgvShopping.CurrentRow.Cells[1].Value.ToString());
7     int Quantity = Int32.Parse(dgvShopping.CurrentRow.Cells[2].Value.ToString());
8     int ProductID = Int32.Parse(dgvShopping.CurrentRow.Cells[3].Value.ToString());
9     //添加调用存储过程命令
10    SqlCommand cmd_DeleteProduction = new SqlCommand("DeleteProduction",
11    this.sqlConnection);
12    cmd_DeleteProduction.CommandType = CommandType.StoredProcedure;
13    //传输数据
14    cmd_DeleteProduction.Parameters.AddWithValue("@ShoppingCarItemID", SqlDbType.Int);
15    cmd_DeleteProduction.Parameters["@ShoppingCarItemID"].Value = ShoppingCartItemID;
16    cmd_DeleteProduction.Parameters.AddWithValue("@ShoppingCarID", SqlDbType.Int);
17    cmd_DeleteProduction.Parameters["@ShoppingCarID"].Value = ShoppingCartID;
18    cmd_DeleteProduction.Parameters.AddWithValue("@Quantity", SqlDbType.Int);
19    cmd_DeleteProduction.Parameters["@Quantity"].Value = Quantity;
20    cmd_DeleteProduction.Parameters.AddWithValue("@ProductID", SqlDbType.Int);
21    cmd_DeleteProduction.Parameters["@ProductID"].Value = ProductID;
22    //打开数据连接
23    this.sqlConnection.Open();
24    cmd_DeleteProduction.ExecuteNonQuery();
25    this.sqlConnection.Close();
26    //重新填充数据库
27    this.dataset.Clear();
28    this.ProductionDataAdapter.Fill(dataset);
29    this.ShoppingDataAdapter.Fill(dataset);
30    //返回提示信息
31    MessageBox.Show("删除成功!", "提示",
32    MessageBoxButtons.OK, MessageBoxIcon.Information
33    );
34 }

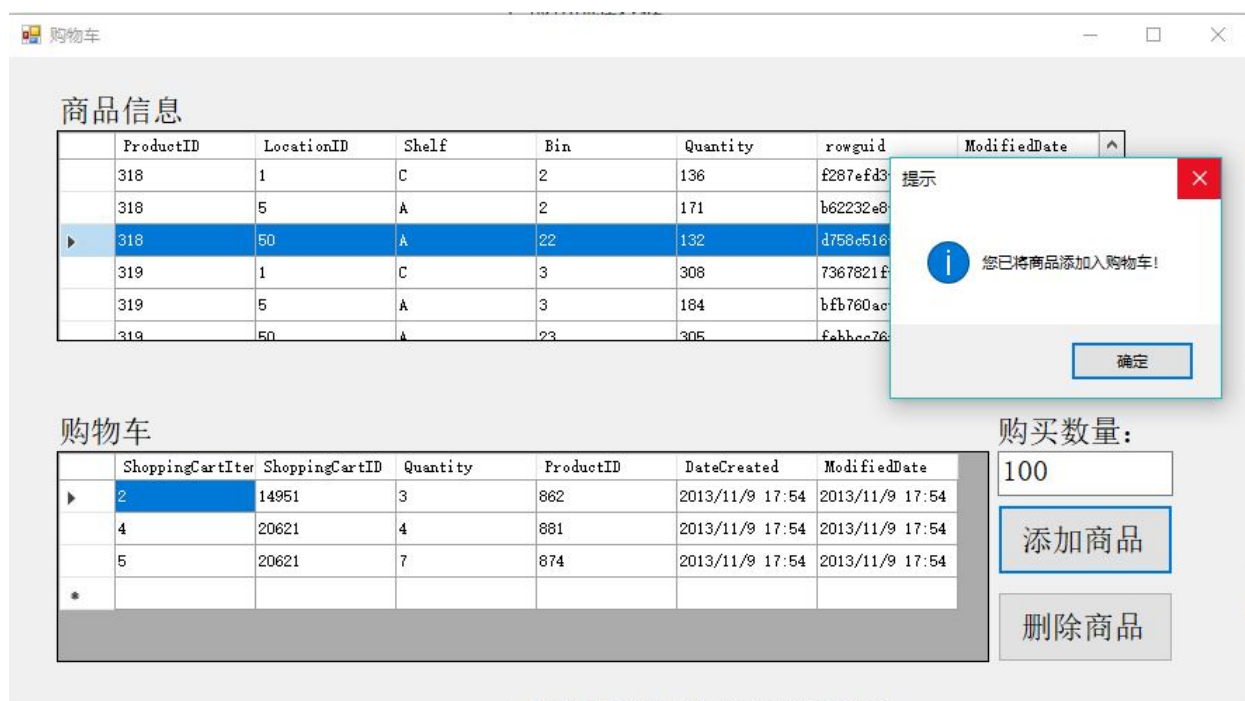
```

二、实验结果

1. 成功创建存储



- 向购物车中添加少于库存数量的商品：在商品信息中点击需要商品，输入商品数量，点击添加商品按钮，弹出提示信息，点击确定，发现购物车中出现新的物品



购物车

商品信息

	ProductID	LocationID	Shelf	Bin	Quantity	rowguid	ModifiedDate
	318	1	C	2	136	f287efd3-ccc...	2014/8/9
	318	5	A	2	171	b62232e8-90b...	2014/8/9
▶	318	50	A	22	32	d758c516-d9b...	2014/8/9
	319	1	C	3	308	7367821f-bb8...	2014/8/9
	319	5	A	3	184	bfb760ac-076...	2014/8/9
	319	50	A	23	305	fabbcc76-276...	2014/8/9

购物车

	ShoppingCartIter	ShoppingCartID	Quantity	ProductID	DateCreated	ModifiedDate
▶	2	14951	3	862	2013/11/9 17:54	2013/11/9 17:54
	4	20621	4	881	2013/11/9 17:54	2013/11/9 17:54
	5	20621	7	874	2013/11/9 17:54	2013/11/9 17:54
	11	50	100	318	2018/9/30 11:04	2018/9/30 11:04
*						

购买数量:

100

添加商品

删除商品

3. 向购物车中添加大于库存数量的商品：与上一步操作相同，但返回错误提示信息

购物车

商品信息

	ProductID	LocationID	Shelf	Bin	Quantity	rowguid	ModifiedDate
▶	318	1	C	2	136	f287efd3-ccc...	2014/8/9
	318	5	A	2	171	b62232e8-90b...	2014/8/9
	318	50				d758c516-d9b...	2014/8/9
	319	1				67821f-bb8...	2014/8/9
	319	5				b760ac-076...	2014/8/9
	319	50				bbcc76-276...	2014/8/9

提示

加入购物车失败！加入购物车数量不得超过库存数量

确定

购物车

	ShoppingCartIter	ShoppingCartID	Quantity	ProductID	DateCreated	ModifiedDate
▶	2	14951	3	862	2013/11/9 17:54	2013/11/9 17:54
	4	20621	4	881	2013/11/9 17:54	2013/11/9 17:54
	5	20621	7	874	2013/11/9 17:54	2013/11/9 17:54
	11	50	100	318	2018/9/30 11:04	2018/9/30 11:04
*						

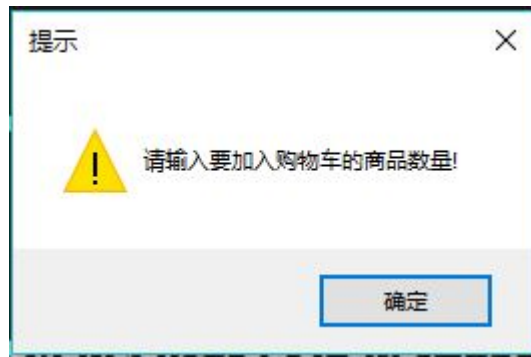
购买数量:

500

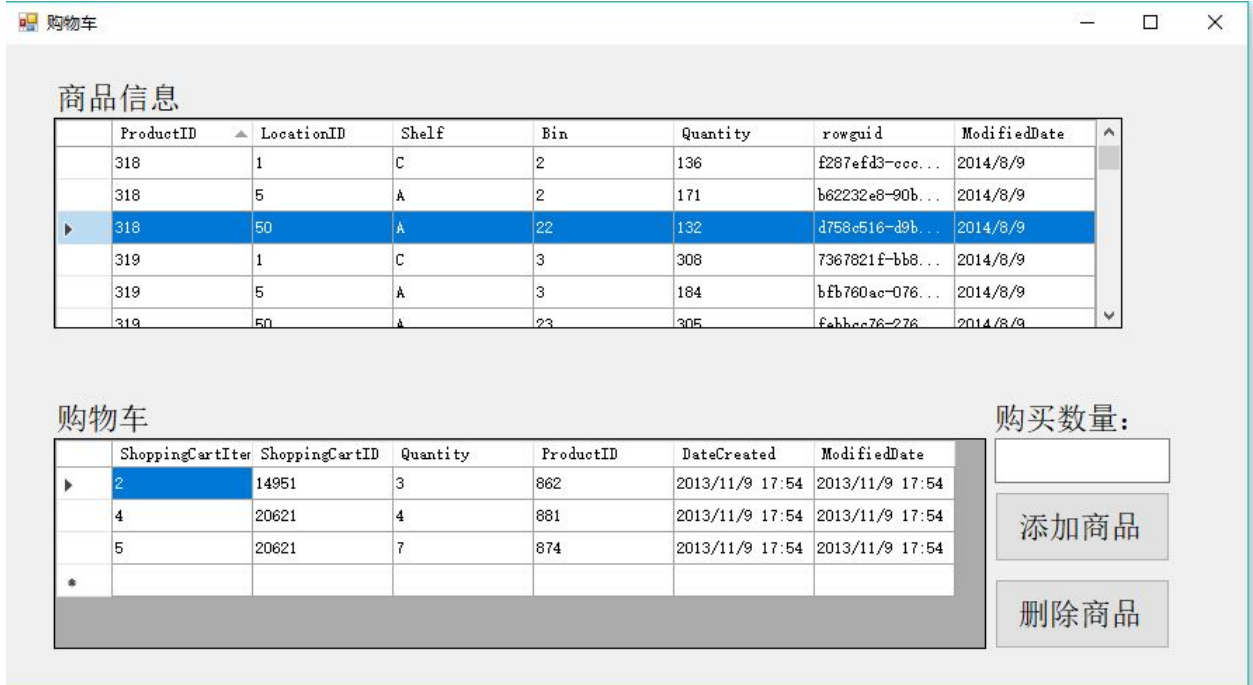
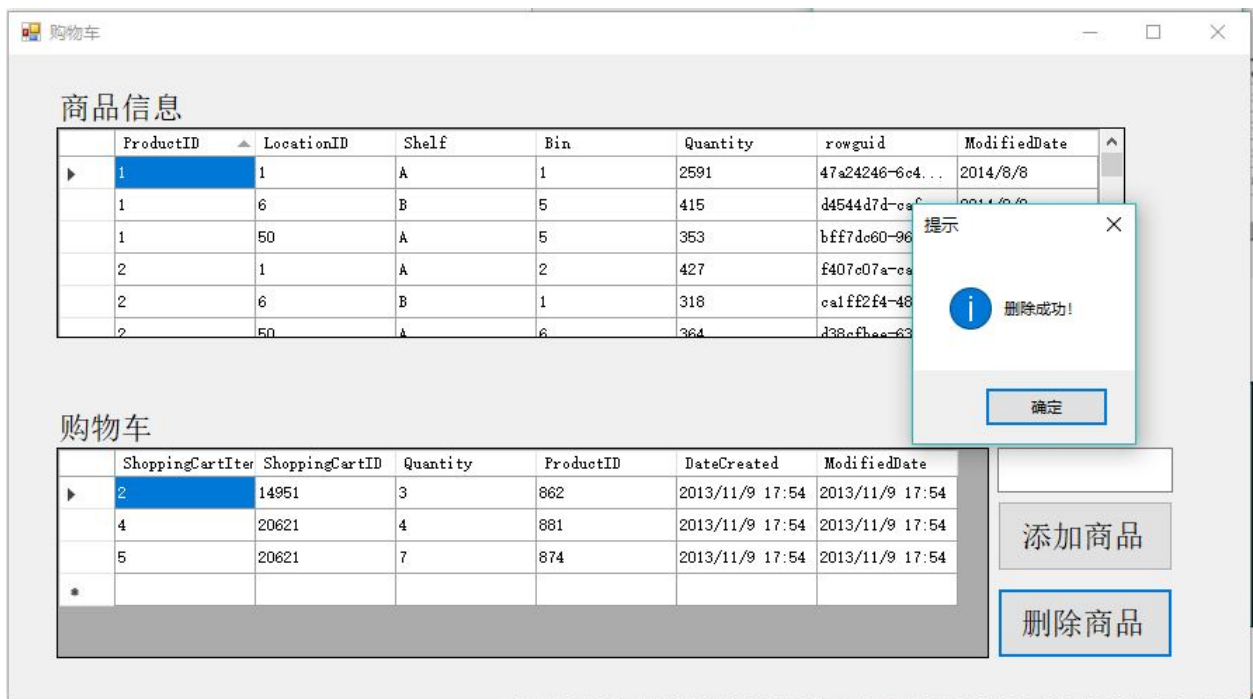
添加商品

删除商品

4. 未输入购买数量就点击添加商品按钮：返回错误信息，提示输入数量



5. 删除商品：选择购物车中商品，点击删除商品按钮，弹出错误信息，商品信息中商品数量增加



三、实验总结

- 刚开始点击添加按钮时，数据未被修改，经检查发现在调用存储过程，传参数是出现问题，修改参数后，成功解决
- 在删除商品时，发现商品信息中商品数量已经恢复，但是购物车中物品为被删除，重写存储过程无果后，上网查找资料后发现，需要在数据集填充前，添加dataset.Clear()函数
- 程序没有设置删除时选择产品数量的功能，可以在之后添加此功能

2018/09/29 19:18

实验人：张启洋

学号：1120161881