ВАРИАНТ 7

Описание предметной области:

Для компании, занимающейся оптовой продажей различных товаров необходимо разработать информационную систему для отслеживания финансовой стороны работы. Деятельность компании организована следующим образом: организация торгует товарами из определенного спектра. Каждый из этих товаров характеризуется ценой, справочной информацией и признаком наличия или отсутствия доставки. В компанию обращаются заказчики. Для каждого из них заполняются стандартные данные (наименование, адрес, телефон, контактное лицо). По каждой сделке составляется документ, в котором хранится информация о покупателе, количестве купленного им товара и дате покупки.

Доставка разных товаров может производиться разными способами, различными по цене и скорости. Нужно хранить информацию по тому, какими способами может осуществляться доставка каждого товара и информацию о том, какой вид доставки (а, соответственно, и какую стоимость доставки) выбрал клиент при заключении сделки.

Выполнить:

1. Изучите предметную область. Создайте инфологическую модель данных и словарь данных для предметной области.

2. Создайте реляционную модель данных предметной области для дальнейшей реализации ИС. Внесите данные в вашу ИС (минимум 5 записей в каждую таблицу).

3. Создайте интерфейс, в котором будет реализован возможность просмотра всех таблиц, редактирования, удаления и добавления информации в них.

4. Создайте 3 запроса с использованием левого или правого внешнего соединения; с использованием внутреннего соединения, на проверку условия через самосоединение, поясните зачем нужно самосоединение.

**1. Инфологическая модель данных предметной области**

Сущности:

1. Товар:

* Название (строка)
* Цена (число)
* Справочная информация (строка)
* Наличие доставки (логическое)

1. Заказчик:

* ID (число)
* Наименование (строка)
* Адрес (строка)
* Телефон (строка)
* Контактное лицо (строка)

1. Способ доставки:

* ID (число)
* Название (строка)
* Цена (число)
* Скорость доставки (строка)

1. Документ (Сделка):

* ID (число)
* Заказчик (связь с сущностью "Заказчик")
* Товар (связь с сущностью "Товар")
* Количество (число)
* Дата покупки (дата)
* Способ доставки (связь с сущностью "Способ доставки")

Связи:

* Один ко многим между "Заказчик" и "Документ"
* Один ко многим между "Товар" и "Документ"
* Один ко многим между "Способ доставки" и "Документ"
* Многие ко многим между "Товар" and "Способ доставки" (реализуется через дополнительную таблицу)

**2. Реляционная модель данных предметной области**

CREATE DATABASE WholesaleTrade;

GO

USE WholesaleTrade;

GO

CREATE TABLE Product (

ProductID INT PRIMARY KEY IDENTITY,

Name VARCHAR(255) NOT NULL,

Price DECIMAL(10, 2) NOT NULL,

Description VARCHAR(MAX),

HasDelivery BIT NOT NULL

);

CREATE TABLE Customer (

CustomerID INT PRIMARY KEY IDENTITY,

Name VARCHAR(255) NOT NULL,

Address VARCHAR(255),

Phone VARCHAR(20),

ContactPerson VARCHAR(255)

);

CREATE TABLE DeliveryMethod (

DeliveryMethodID INT PRIMARY KEY IDENTITY,

Name VARCHAR(255) NOT NULL,

Price DECIMAL(10, 2) NOT NULL,

Speed VARCHAR(255)

);

CREATE TABLE OrderDocument (

DocumentID INT PRIMARY KEY IDENTITY,

CustomerID INT NOT NULL,

ProductID INT NOT NULL,

Quantity INT NOT NULL,

PurchaseDate DATE NOT NULL,

DeliveryMethodID INT,

FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID),

FOREIGN KEY (ProductID) REFERENCES Product(ProductID),

FOREIGN KEY (DeliveryMethodID) REFERENCES DeliveryMethod(DeliveryMethodID)

);

CREATE TABLE ProductDelivery (

ProductID INT NOT NULL,

DeliveryMethodID INT NOT NULL,

PRIMARY KEY (ProductID, DeliveryMethodID),

FOREIGN KEY (ProductID) REFERENCES Product(ProductID),

FOREIGN KEY (DeliveryMethodID) REFERENCES DeliveryMethod(DeliveryMethodID)

);

INSERT INTO Product (Name, Price, Description, HasDelivery) VALUES

('Ноутбук', 50000, 'Мощный игровой ноутбук', 1),

('Клавиатура', 2000, 'Механическая клавиатура', 1),

('Мышь', 2000, 'Беспроводная мышь', 1),

('Монитор', 15000, 'Монитор с диагональю 27 дюймов', 1),

('Принтер', 7000, 'Лазерный принтер', 1);

INSERT INTO Customer (Name, Address, Phone, ContactPerson) VALUES

('ООО "Ромашка"', 'ул. Ленина, 1', '88005553535', 'Иванов Иван'),

('ИП Петров', 'ул. Пушкина, 2', '89123456789', 'Петров Петр'),

('Сидоров и Ко', 'ул. Советская, 3', '84951234567', 'Сидоров Сидор'),

('Магазин "Компьютеры"', 'пр. Победы, 4', '89012345678', 'Кузнецов Кузьма'),

('Офис "Техника"', 'ул. Мира, 5', '89991234567', 'Попов Павел');

INSERT INTO DeliveryMethod (Name, Price, Speed) VALUES

('Курьерская доставка', 500, '1-2 дня'),

('Самовывоз', 0, 'В день заказа'),

('Почта России', 300, '7-14 дней'),

('Транспортная компания', 1000, '3-5 дней'),

('Экспресс-доставка', 1500, 'В течение дня');

INSERT INTO OrderDocument (CustomerID, ProductID, Quantity, PurchaseDate, DeliveryMethodID) VALUES

(1, 1, 10, '2023-06-20', 1),

(2, 2, 5, '2023-06-21', 2),

(3, 3, 20, '2023-06-22', 3),

(4, 4, 15, '2023-06-23', 4),

(5, 5, 8, '2023-06-24', 5);

INSERT INTO ProductDelivery (ProductID, DeliveryMethodID) VALUES

(1, 1), (1, 2), (1, 4),

(2, 1), (2, 3),

(3, 2), (3, 5),

(4, 1), (4, 2), (4, 3), (4, 4),

(5, 1), (5, 5);

**3. Интерфейс**

**Form1.cs:**  
using System;

using System.Data;

using System.Data.SqlClient;

using System.Windows.Forms;

namespace WholesaleTradeApp

{

public partial class Form1 : Form

{

private SqlConnection connection;

private SqlDataAdapter adapterProduct, adapterCustomer, adapterDeliveryMethod, adapterOrderDocument, adapterProductDelivery;

private DataSet dataSet;

private string connectionString = @"Data Source=HOME-PC\SQLEXPRESS;Initial Catalog=WholesaleTrade;Integrated Security=True";

public Form1()

{

InitializeComponent();

InitializeDatabaseConnection();

LoadData();

}

private void InitializeDatabaseConnection()

{

connection = new SqlConnection(connectionString);

}

private void LoadData()

{

try

{

connection.Open();

dataSet = new DataSet();

adapterProduct = new SqlDataAdapter("SELECT \* FROM Product", connection);

adapterProduct.Fill(dataSet, "Product");

adapterCustomer = new SqlDataAdapter("SELECT \* FROM Customer", connection);

adapterCustomer.Fill(dataSet, "Customer");

adapterDeliveryMethod = new SqlDataAdapter("SELECT \* FROM DeliveryMethod", connection);

adapterDeliveryMethod.Fill(dataSet, "DeliveryMethod");

adapterOrderDocument = new SqlDataAdapter("SELECT \* FROM OrderDocument", connection);

adapterOrderDocument.Fill(dataSet, "OrderDocument");

adapterProductDelivery = new SqlDataAdapter("SELECT \* FROM ProductDelivery", connection);

adapterProductDelivery.Fill(dataSet, "ProductDelivery");

dataGridView1.DataSource = dataSet.Tables["Product"];

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных: " + ex.Message);

}

finally

{

connection.Close();

}

}

private void btnSave\_Click(object sender, EventArgs e)

{

try

{

connection.Open();

SqlCommandBuilder commandBuilderProduct = new SqlCommandBuilder(adapterProduct);

adapterProduct.Update(dataSet, "Product");

SqlCommandBuilder commandBuilderCustomer = new SqlCommandBuilder(adapterCustomer);

adapterCustomer.Update(dataSet, "Customer");

SqlCommandBuilder commandBuilderDeliveryMethod = new SqlCommandBuilder(adapterDeliveryMethod);

adapterDeliveryMethod.Update(dataSet, "DeliveryMethod");

SqlCommandBuilder commandBuilderOrderDocument = new SqlCommandBuilder(adapterOrderDocument);

adapterOrderDocument.Update(dataSet, "OrderDocument");

SqlCommandBuilder commandBuilderProductDelivery = new SqlCommandBuilder(adapterProductDelivery);

adapterProductDelivery.Update(dataSet, "ProductDelivery");

MessageBox.Show("Данные успешно сохранены!");

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при сохранении данных: " + ex.Message);

}

finally

{

connection.Close();

}

}

private void btnAdd\_Click(object sender, EventArgs e)

{

DataRow newRow = dataSet.Tables[dataGridView1.DataSource.ToString()].NewRow();

dataSet.Tables[dataGridView1.DataSource.ToString()].Rows.Add(newRow);

}

private void btnDelete\_Click(object sender, EventArgs e)

{

foreach (DataGridViewRow row in dataGridView1.SelectedRows)

{

dataGridView1.Rows.RemoveAt(row.Index);

}

}

private void comboBoxTables\_SelectedIndexChanged(object sender, EventArgs e)

{

dataGridView1.DataSource = dataSet.Tables[comboBoxTables.SelectedItem.ToString()];

}

}

}

**Form1.Designer.cs:**  
namespace WholesaleTradeApp

{

partial class Form1

{

private System.ComponentModel.IContainer components = null;

private System.Windows.Forms.DataGridView dataGridView1;

private System.Windows.Forms.Button btnSave;

private System.Windows.Forms.Button btnAdd;

private System.Windows.Forms.Button btnDelete;

private System.Windows.Forms.ComboBox comboBoxTables;

private System.Windows.Forms.Label labelTable;

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

private void InitializeComponent()

{

this.dataGridView1 = new System.Windows.Forms.DataGridView();

this.btnSave = new System.Windows.Forms.Button();

this.btnAdd = new System.Windows.Forms.Button();

this.btnDelete = new System.Windows.Forms.Button();

this.comboBoxTables = new System.Windows.Forms.ComboBox();

this.labelTable = new System.Windows.Forms.Label();

((System.ComponentModel.ISupportInitialize)(this.dataGridView1)).BeginInit();

this.SuspendLayout();

//

// dataGridView1

//

this.dataGridView1.ColumnHeadersHeightSizeMode = System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;

this.dataGridView1.Location = new System.Drawing.Point(12, 41);

this.dataGridView1.Name = "dataGridView1";

this.dataGridView1.Size = new System.Drawing.Size(776, 386);

this.dataGridView1.TabIndex = 0;

//

// btnSave

//

this.btnSave.Location = new System.Drawing.Point(12, 433);

this.btnSave.Name = "btnSave";

this.btnSave.Size = new System.Drawing.Size(75, 23);

this.btnSave.TabIndex = 1;

this.btnSave.Text = "Сохранить";

this.btnSave.UseVisualStyleBackColor = true;

this.btnSave.Click += new System.EventHandler(this.btnSave\_Click);

//

// btnAdd

//

this.btnAdd.Location = new System.Drawing.Point(93, 433);

this.btnAdd.Name = "btnAdd";

this.btnAdd.Size = new System.Drawing.Size(75, 23);

this.btnAdd.TabIndex = 2;

this.btnAdd.Text = "Добавить";

this.btnAdd.UseVisualStyleBackColor = true;

this.btnAdd.Click += new System.EventHandler(this.btnAdd\_Click);

//

// btnDelete

//

this.btnDelete.Location = new System.Drawing.Point(174, 433);

this.btnDelete.Name = "btnDelete";

this.btnDelete.Size = new System.Drawing.Size(75, 23);

this.btnDelete.TabIndex = 3;

this.btnDelete.Text = "Удалить";

this.btnDelete.UseVisualStyleBackColor = true;

this.btnDelete.Click += new System.EventHandler(this.btnDelete\_Click);

//

// comboBoxTables

//

this.comboBoxTables.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;

this.comboBoxTables.FormattingEnabled = true;

this.comboBoxTables.Items.AddRange(new object[] {

"Product",

"Customer",

"DeliveryMethod",

"OrderDocument",

"ProductDelivery"});

this.comboBoxTables.Location = new System.Drawing.Point(667, 12);

this.comboBoxTables.Name = "comboBoxTables";

this.comboBoxTables.Size = new System.Drawing.Size(121, 21);

this.comboBoxTables.TabIndex = 4;

this.comboBoxTables.SelectedIndexChanged += new System.EventHandler(this.comboBoxTables\_SelectedIndexChanged);

//

// labelTable

//

this.labelTable.AutoSize = true;

this.labelTable.Location = new System.Drawing.Point(614, 15);

this.labelTable.Name = "labelTable";

this.labelTable.Size = new System.Drawing.Size(47, 13);

this.labelTable.TabIndex = 5;

this.labelTable.Text = "Таблица";

//

// Form1

//

this.ClientSize = new System.Drawing.Size(800, 468);

this.Controls.Add(this.labelTable);

this.Controls.Add(this.comboBoxTables);

this.Controls.Add(this.btnDelete);

this.Controls.Add(this.btnAdd);

this.Controls.Add(this.btnSave);

this.Controls.Add(this.dataGridView1);

this.Name = "Form1";

this.Text = "Оптовая торговля";

((System.ComponentModel.ISupportInitialize)(this.dataGridView1)).EndInit();

this.ResumeLayout(false);

this.PerformLayout();

}

}

}

**4. Запросы к БД**

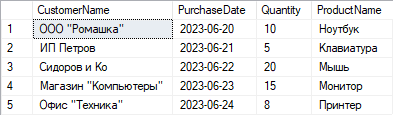
**Запрос с использованием левое внешнего соединения**

SELECT c.Name AS CustomerName, o.PurchaseDate, o.Quantity, p.Name AS ProductName

FROM Customer c

LEFT JOIN OrderDocument o ON c.CustomerID = o.CustomerID

LEFT JOIN Product p ON o.ProductID = p.ProductID; Client;



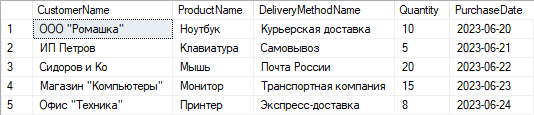
**Запрос с использованием внутреннего соединения**

SELECT c.Name AS CustomerName, p.Name AS ProductName, d.Name AS DeliveryMethodName, o.Quantity, o.PurchaseDate

FROM OrderDocument o

INNER JOIN Customer c ON o.CustomerID = c.CustomerID

INNER JOIN Product p ON o.ProductID = p.ProductID

INNER JOIN DeliveryMethod d ON o.DeliveryMethodID = d.DeliveryMethodID; 

**Запрос с использованием самосоединения**

SELECT p1.Name, p1.Price

FROM Product p1

INNER JOIN Product p2 ON p1.Price = p2.Price AND p1.ProductID != p2.ProductID; RoomID IN (SELECT RoomID FROM Room WHERE RoomType = 'Suite');



**Зачем нужно самосоединение?**

Самосоединение используется для сравнения строк одной и той же таблицы.