МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ  
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»  
Кафедра «Системний аналіз та інформаційно-аналітичні технології»

КУРСОВА РОБОТА

з курсу «Проектування баз даних та інформаційних систем»

Тема: «Розробка прикладної програми графічного інтерфейсу користувача для магазину побутової техніки»

Керівник роботи:

к.т.н., доцент каф. САІТ Коваленко С. В.

Виконав:

студент групи КН-320б Миргород В. І.

Харків 2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ  
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»  
Кафедра «Системний аналіз та інформаційно-аналітичні технології»

Студент – Миргород В. І. Група – КН-320б Курс – 3

ЗАВДАННЯ

на науково-дослідну курсову роботу

з курсу «Проектування баз даних та інформаційних систем»

Тема: «Розробка прикладної програми графічного інтерфейсу користувача для магазину побутової техніки»

Постановка задачі: спроектувати та реалізувати базу даних і програму графічного інтерфейсу користувача для магазину побутової техніки.

Графічний інтерфейс користувача передбачає програмний додаток з можливістю огляду, пошуку, додавання, видалення та змінення даних в розробленій базі.

Реалізація програми повинна здійснюватися з використанням об’єктно-орієнтованих технологій. Слід представити необхідну інформацію про програму та способи її застосування.

Короткий зміст роботи:

1. Реферативна частина

Дослідження поставленої задачі, вибір на обгрунтування необхідних програмних методів розв’язання задачі.

1. Теоретична частина

Визначення вимог щодо програмного забезпечення

1. Програмна частина

Реалізація тексту програми, консольного тексту, тестування роботи програми. Документація програмного інтерфейсу.

ЗМІСТ

[ВСТУП 4](#_Toc120516576)

[1 ПОСТАНОВКА ЗАДАЧІ 5](#_Toc120516577)

[2 ОСНОВИ РЕЛЯЦІЙНИХ БАЗ ДАНИХ 6](#_Toc120516578)

[3 ПРОЕКТУВАННЯ БАЗ ДАНИХ 10](#_Toc120516579)

[4 КЕРІВНИЦТВО ТА ТЕСТОВІ ПРИКЛАДИ 14](#_Toc120516580)

[ВИСНОВКИ 18](#_Toc120516581)

[СПИСОК ЛІТЕРАТУРИ 19](#_Toc120516582)

[ДОДАТОК 20](#_Toc120516583)

# ВСТУП

В наш час покупку в інтернеті – дуже популярний та зручний спосіб придбати будь-яку річ. Будь-який користувач може легко знайти річ, яку він хоче придбати, в інтернеті.

Тому будь-який сучасний магазин має свій сайт або спеціальний програмний інтерфейс, який дозволяє швидко та легко знайти та придбати будь-яку річ, що продається в цьому магазині.

Важливо, щоб інтерфейс програми був якомога простішим і зрозумілішим для користувача, навіть якщо йому складно орієнтуватися в комп’ютері та інтернеті.

В таких програмах можна отримати будь-яку потрібну інформацію про товар, його характеристики, наявність і таке інше. Часто в таких інтернет-магазинах можна оформити покупку, вказати свої особисті дані, та через деякий час отримати своє замовлення, навіть не виходячи з дому. В таких магазинах можна зручно порівняти товари між собою за характеристиками, і обрати той варіант, що найбільше підходить покупцеві.

Важливо, щоб така програма чи інтернет-сайт надавала всі умови для легкого та зручного користування, тому під час розробки програми потрібно звернути на це увагу.

# 1 ПОСТАНОВКА ЗАДАЧІ

У даній роботі потрібно спроектувати та реалізувати базу даних та прикладну програму графічного інтерфейсу користувача для магазину побутових товарів.

У програмі потрібно передбачити пошук інформації, додавання, видалення та змінення даних.

Реалізація програми повинна здійснюватися з використанням об’єктно-орієнтованих технологій. Слід представити необхідну інформацію про програму та способи її застосування.

Актуальність даної програми полягає в тому, що в багатьох комерційних системах потрібно зберігати велику кількість інформації, що стосується товарів, клієнтів, продажів і таке інше. Таку інформацію зручно зберігати розподілено, у різних таблицях, але у вигляді однієї загальної бази даних.

Для цього потрібно виконати наступні пункти:

1. Створення бази даних
2. Створення необхідних тригерів і представлень
3. Підключення бази даних до програми
4. Створення необхідного інтерфейсу, що дозволить взаємодіяти з базою даних
5. Тестування програми за допомогою початкових даних, заздалегідь внесених до бази даних

Після виконання цих дій, потрібно також розробити технічну документацію до програми, в якій роз’яснити способи та особливості використання основних дій. Також привести код програми.

# 2 ОСНОВИ РЕЛЯЦІЙНИХ БАЗ ДАНИХ

База даних– сукупність даних, організованих відповідно до концепції, яка описує характеристику цих даних і взаємозв'язки між їх елементами; ця сукупність підтримує щонайменше одну з областей застосування. В загальному випадку база даних містить схеми, таблиці, подання, збережені процедури та інші об'єкти.

Дані у базі організовують відповідно до моделі організації даних. Таким чином, сучасна база даних, крім самих даних, містить їх опис та може містити засоби для їх обробки.

Реляційна база даних – база даних, заснована на реляційній моделі даних. Слово «реляційний» походить від англ. *relation* (відношення). Для роботи з реляційними БД застосовують реляційні СКБД. Інакше кажучи, реляційна база даних – це база даних, яка сприймається користувачем як набір нормалізованих відношень різного ступеня.

Реляційна база даних є сукупністю елементів даних, організованих у вигляді набору формально описаних таблиць, з яких дані можуть бути доступними або повторно зібрані багатьма різними способами без необхідності реорганізації таблиць бази даних.

Конкретні дані в реляційній базі даних визначаються за допомогою первинного ключа. Первинний ключ – атрибут або набір атрибутів, який однозначно ідентифікує кортеж даного відношення. Первинний ключ обов'язково унікальний. У реляційних базах даних первинний ключ обирають серед одного з потенційних ключів або ж генерують сурогатний ключ.

Ще одне важливе визначення в реляційних базах даних – зовнішній ключ. Зовнішній ключ – атрибут (набір атрибутів) в деякому відношенні R, який відповідає первинному ключу іншого відношення або того ж таки відношення R.

2.1 Нормальні форми

Нормальна форма — формальна властивість відношення, яка характеризує ступінь надмірності збережуваних даних і можливі проблеми. Кожна наступна нормальна форма в нижченаведеному списку (крім ДКНФ) в деякому сенсі є досконалішою, ніж попередня, з точки зору усунення надмірності.

Перша нормальна форма (1НФ, 1NF) це властивість відношення у реляційній баз даних. Відношення знаходиться в першій нормальній формі тоді і тільки тоді, коли домен кожного атрибута містить лише нероздільні значення, а значення кожного атрибута містить лише одне значення з цього домену. Критерії першої нормальної форми:

* Кожна таблиця повинна мати основний ключ: мінімальний набір колонок, які ідентифікують запис.
* Уникнення повторень груп (категорії даних, що можуть зустрічатись різну кількість раз в різних записах) правильно визначаючи неключові атрибути.
* Атомарність: кожен атрибут повинен мати лише одне значення, а не множину значень.

Друга нормальна форма (2НФ, 2NF) — нормальна форма, що використовується для нормалізації баз даних. 2НФ первісно була визначена 1971 року Едгаром Коддом. Щоб перебувати в другій нормальній формі, таблиця, що перебуває в першій нормальній формі, має відповідати додатковим критеріям. А саме: 1НФ таблиця перебуватиме в 2НФ тоді й лише тоді, коли для будь-якого потенційного ключа K і будь-якого атрибута A, який не є частиною потенційного ключа, A залежить саме від цілого потенційного ключа, а не від його частини.

Тобто, 1НФ таблиця перебуває в 2НФ тоді й тільки тоді, коли всі її неключові атрибути функціонально залежні від потенційного ключа в цілому.

У разі, якщо 1НФ таблиця не має складних потенційних ключів (таких, що складаються більш ніж з одного атрибута), тоді вона автоматично перебуватиме в 2НФ.

Третя нормальна форма (3НФ) — нормальна форма використовна в нормалізації баз даних. 3НФ первісно була визначена 1971 року Едгаром Коддом. За Коддом таблиця знаходиться в 3НФ тоді й лише тоді, коли виконуються наступні умови:

* Відношення R (таблиця) знаходиться в 2НФ
* Кожен неключовий атрибут відношення R нетранзитивно (безпосередньо) залежить від кожного потенційного ключа в R.

|  |
| --- |
| Id |
| Id\_товара |
| Бренд |
| Модель |
| Країна |

Одна з таблиць – характеристика товару, створена протягом написання бази даних для курсового проекту. Вона знаходиться в 3НФ, бо відповідає вимогам 2НФ (особистий ключ, немає повторень полів, дані, що повторюються, винесені в окрему таблицю), а також не має полів, що залежать від основного ключа, або від іншого поля.

2.2 Представлення

Представлення - це віртуальна таблиця, вміст якої визначається запитом. Як і таблиця, представлення складається з низки іменованих стовпців і рядків даних. Поки представлення не буде проіндексовано, воно не існує в базі даних як збережена сукупність значень. Рядки і стовпці даних витягуються з таблиць, зазначених у запиті, що визначає представлення, і динамічно створюваних під час звернень до представлення.

Представлення виконує функцію фільтра базових таблиць, на які воно посилається. Запит, що визначає представлення, може бути ініційований в одній або декількох таблицях або в інших представленнях поточної або інших баз даних. Крім того, для визначення представлень із даними з декількох різнорідних джерел можна використовувати розподілені запити. Це корисно, наприклад, якщо потрібно об'єднати структуровані подібним чином дані, які стосуються різних серверів, кожен з яких зберігає дані конкретного відділу організації.

Представлення створюється за допомогою команд

Create view [Назва представлення] as [Запит select]

Приклад створення представлення для виведення інформації про чеки та продажі:

create view view\_sales

as

select sale.id as 'Номер чека', sale.date\_sale as 'Дата', info.brand as 'Производитель', info.model as 'Модель', sale\_row.k as 'Количество', product.price as 'Цена'

from sale join sale\_row on sale\_row.id\_sale = sale.id join product on sale\_row.id\_product = product.id join info on info.id\_product = product.id

go

2.3 Тригери

Тригер - це особливий різновид збереженої процедури, яка автоматично виконується при виникненні події на сервері бази даних. Тригери DML виконуються, коли користувач намагається змінити дані за допомогою подій мови обробки даних (DML). Подіями DML є процедури INSERT, UPDATE або DELETE, що застосовуються до таблиці або подання. Ці тригери спрацьовують під час запуску будь-якої допустимої події незалежно від наявності та кількості порушених рядків таблиці.

Тригери DDL активуються у відповідь на різні події мови опису даних (DDL). Ці події насамперед відповідають інструкціям Transact-SQL CREATE, ALTER, DROP і деяким системним збереженим процедурам, які виконують схожі з DDL операції.

Тригер створюється за допомогою команд

Create trigger [Назва тригеру] as begin [Запити] end

Приклад створення тригеру для видалення інформації про чеки (продажі):

create trigger tr1

on sale

instead of delete

as

begin

declare @id int = (select top 1 id from deleted)

delete from sale\_row where id\_sale = @id

delete from sale where id = @id

end

go

# 3 ПРОЕКТУВАННЯ БАЗ ДАНИХ

Створення бази даних починається з проектування. В результаті проектування потрібно отримати готову структуру таблиць бази даних із зв’язками. Структура визначається складом стовпців, типом даних, що зберігаються у стовпцях, а також ключем для таблиці.

3.1 Інфологічна модель

Інфологічна модель відображає інформацію про предметну область у вигляді, незалежному від СУБД, що використовується. Ця модель відображає інформаційно-логічний рівень абстрагування, який пов’язаний з описом об’єктів предметної області, їх властивостей та взаємозв’язків.

Часто ці моделі ототожнюють з концептуальними моделями предметної області і називають концептуальними інфологічними моделями (внутрішня і зовнішня концептуальні інфологічні моделі).

Приклад моделі для бази даних обліку оренди житла:

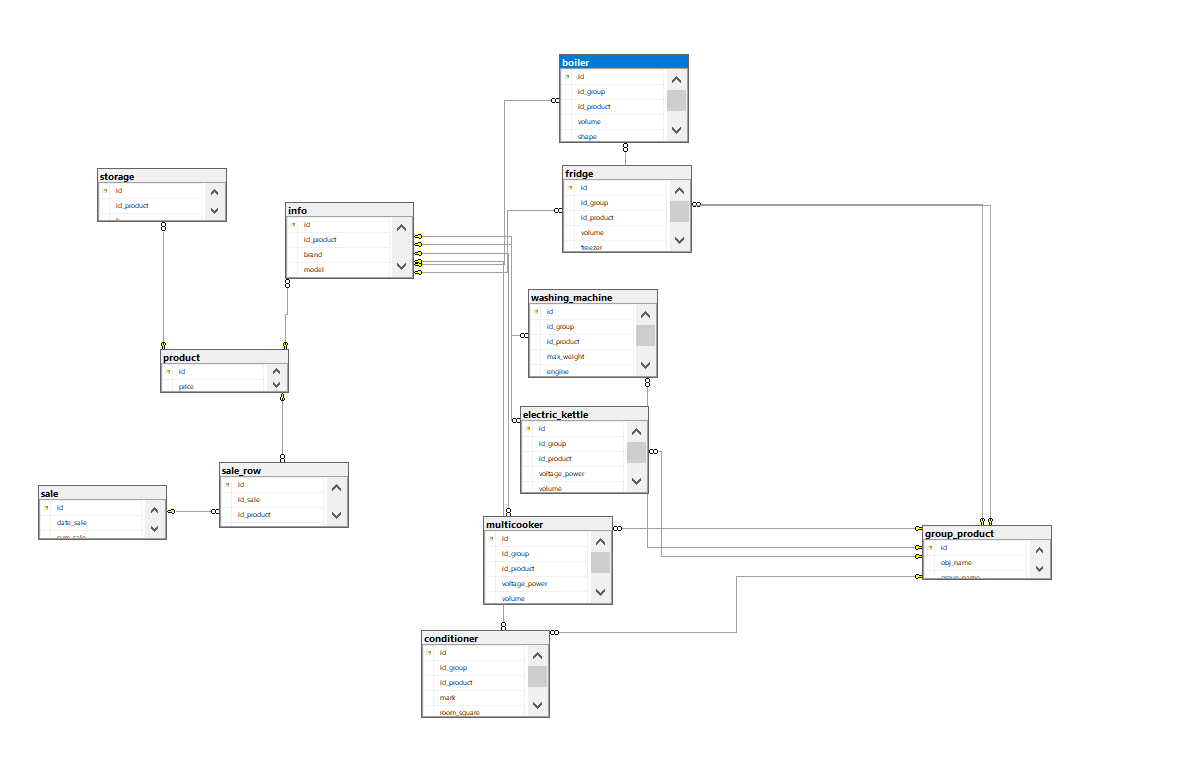


Рисунок 3‑1 – Модель бази даних

3.2 Даталогічна модель

Даталогічна модель – модель логічного рівня, яка відображає логічні зв’язки між елементами даних незалежно від їх змісту і середовища збереження. Часто ці моделі ототожнюють з логічними моделями.

Приклади таблиць в базі даних системи оренди житла:

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Тип | Призначення | Особливості |
| Id | Int | Id | Primary key |
| Price | int | Ціна |  |

Таблица 3.1 – Товар

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Тип | Призначення | Особливості |
| Id | Int | Id | Primary key |
| Date\_sale | date | Дата продажу |  |
| Sum\_sale | int | Сума продажу |  |

Таблица 3.2 - Чек

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Тип | Призначення | Особливості |
| Id | Int | Id | Primary key |
| Id\_sale | int | Id чеку | Foreign key |
| Id\_product | int | Id товару | Foreign key |
| K | int | Кількість |  |

Таблица 3.3 – Рядок чеку

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Тип | Призначення | Особливості |
| Id | Int | Id | Primary key |
| Id\_product | int | Id товару | Foreign key |
| K | int | Кількість |  |

Таблица 3.4 – Склад

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Тип | Призначення | Особливості |
| Id | Int | Id | Primary key |
| Id\_product | int | Id товару | Foreign key |
| Brand | Nvarchar(50) | Бренд |  |
| Model | Nvarchar(50) | Модель |  |
| Country | Nvarchar(50) | Країна |  |

Таблица 3.5 – Характеристики

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Тип | Призначення | Особливості |
| Id | Int | Id | Primary key |
| Obj\_name | Nvarchar(50) | Підкатегорія |  |
| Group\_name | Nvarchar(50) | Група товарів |  |

Таблица 3.6 – Група товарів

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Тип | Призначення | Особливості |
| Id | Int | Id | Primary key |
| Id\_group | int | Id групи | Foreign key |
| Id\_product | int | Id товару | Foreign key |
| Volume | int | Об’єм |  |
| Shape | Nvarchar(50) | Форма |  |
| Heater | Nvarchar(50) | Нагрівач |  |

Таблица 3.7 –Бойлер

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Тип | Призначення | Особливості |
| Id | Int | Id | Primary key |
| Id\_group | int | Id групи | Foreign key |
| Id\_product | int | Id товару | Foreign key |
| mark | int | Маркування |  |
| Room\_square | int | Площа роботи |  |
| Kind | Nvarchar(50) | Тип |  |

Таблица 3.8 –Кондиціонер

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Тип | Призначення | Особливості |
| Id | Int | Id | Primary key |
| Id\_group | int | Id групи | Foreign key |
| Id\_product | int | Id товару | Foreign key |
| Voltage\_power | int | Потужність |  |
| volume | int | Об’єм |  |
| material | Nvarchar(50) | Матеріал |  |

Таблица 3.9 –Електрочайник

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Тип | Призначення | Особливості |
| Id | Int | Id | Primary key |
| Id\_group | int | Id групи | Foreign key |
| Id\_product | int | Id товару | Foreign key |
| volume | int | Об’єм |  |
| freezer | Nvarchar(50) | Морозилка |  |
| compressor | Nvarchar(50) | Компресор |  |

Таблица 3.10 –Холодильник

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Тип | Призначення | Особливості |
| Id | Int | Id | Primary key |
| Id\_group | int | Id групи | Foreign key |
| Id\_product | int | Id товару | Foreign key |
| Voltage\_power | int | Потужність |  |
| volume | int | Об’єм |  |
| kind | Nvarchar(50) | Тип |  |

Таблица 3.11 –Мультиварка

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Тип | Призначення | Особливості |
| Id | Int | Id | Primary key |
| Id\_group | int | Id групи | Foreign key |
| Id\_product | int | Id товару | Foreign key |
| Max\_weight | int | Макс вага |  |
| engine | Nvarchar(50) | Тип двигуна |  |
| loading | Nvarchar(50) | Тип завантаж. |  |

Таблица 3.12 –Пральна машина

# 4 КЕРІВНИЦТВО ТА ТЕСТОВІ ПРИКЛАДИ

ADO.NET являє собою технологію роботи з даними, яка заснована на платформі .NET Framework. Ця технологія представляє нам набір класів, через які ми можемо надсилати запити до баз даних, встановлювати з'єднання, отримувати відповідь від бази даних і виконувати низку інших операцій.

Причому важливо зазначити, що систем управління баз даних може бути безліч. За своєю сутністю вони можуть відрізнятися. MS SQL Server, наприклад, для створення запитів використовує мову T-SQL, а MySQL і Oracle застосовують мову PL-SQL. Різні системи баз даних можуть мати різні типи даних. Також можуть відрізнятися якісь інші моменти. Однак функціонал ADO.NET побудований таким чином, щоб надати розробникам уніфікований інтерфейс для роботи з найрізноманітнішими СУБД.

Основу інтерфейсу взаємодії з базами даних в ADO.NET становить обмежене коло об'єктів: Connection, Command, DataReader, DataSet і DataAdapter. За допомогою об'єкта Connection відбувається встановлення підключення до джерела даних. Об'єкт Command дає змогу виконувати операції з даними з БД. Об'єкт DataReader зчитує отримані в результаті запиту дані. Об'єкт DataSet призначений для зберігання даних з БД і дозволяє працювати з ними незалежно від БД. І об'єкт DataAdapter є посередником між DataSet і джерелом даних. Головним чином, через ці об'єкти і буде йти робота з базою даних.

Приклад підключення до бази даних за допомогою ADO.NET

string cons = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\coursework\_shop.mdf;Integrated Security=True;Connect Timeout=30";

SqlConnection con = new SqlConnection(cons);

con.Open();

con.Close();

Після підключення до бази даних ми можемо написати і виконати запит на мові sql.

Тестування:

Приклад пошуку товару в магазині із застосуванням фільтру.

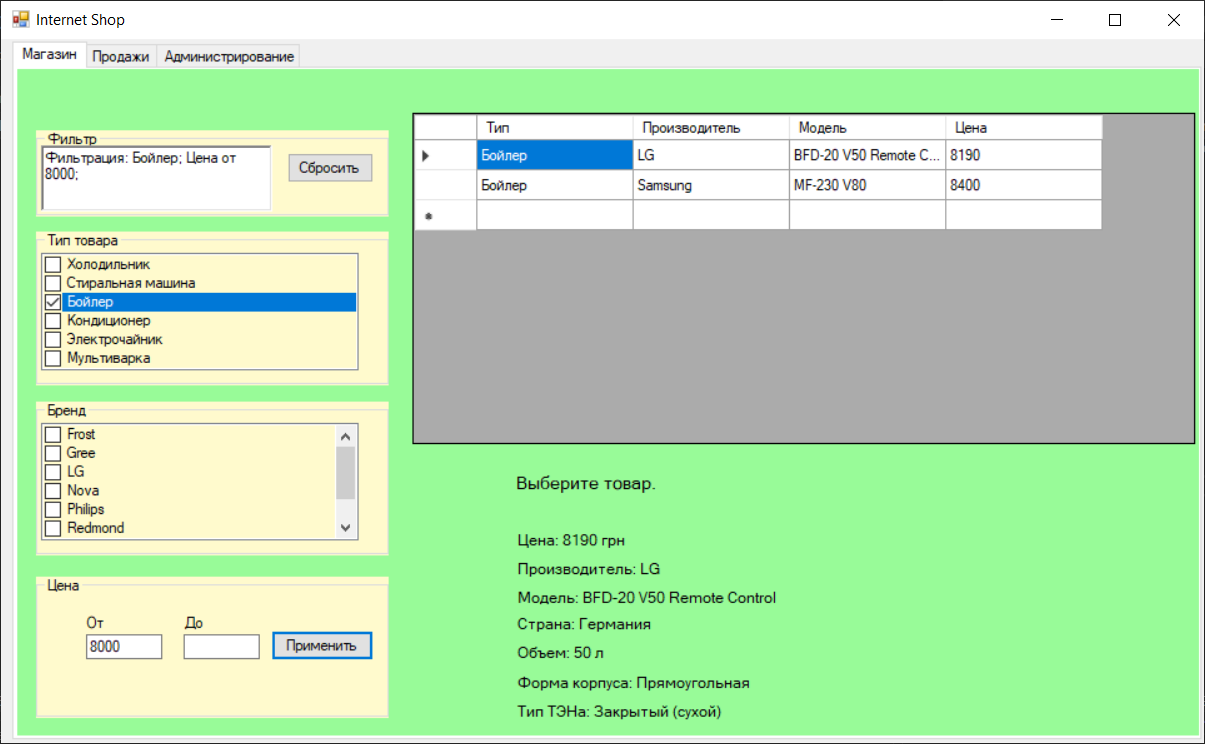


Рисунок 4‑1 – Пошук

Приклад додавання та видалення чеку з системи обліку продажів:

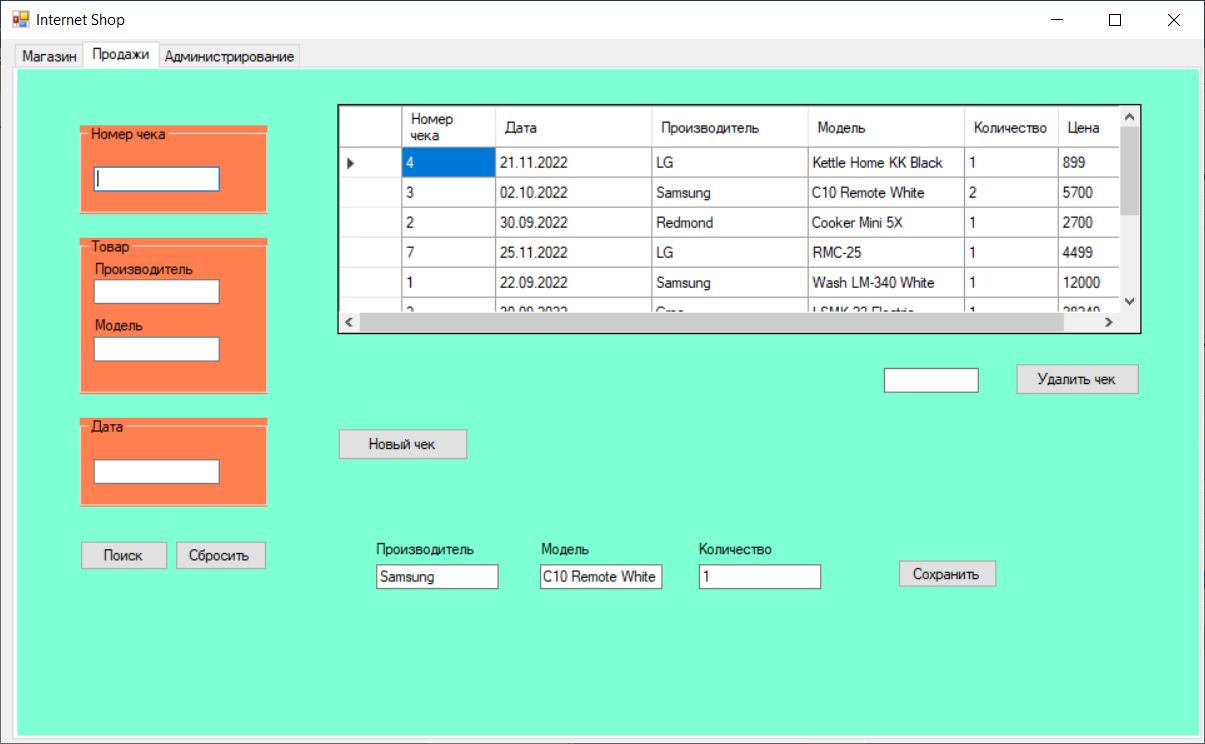


Рисунок 4‑2 – Додавання, змінення, видалення

Приклад зміни характеристик деякого товару:

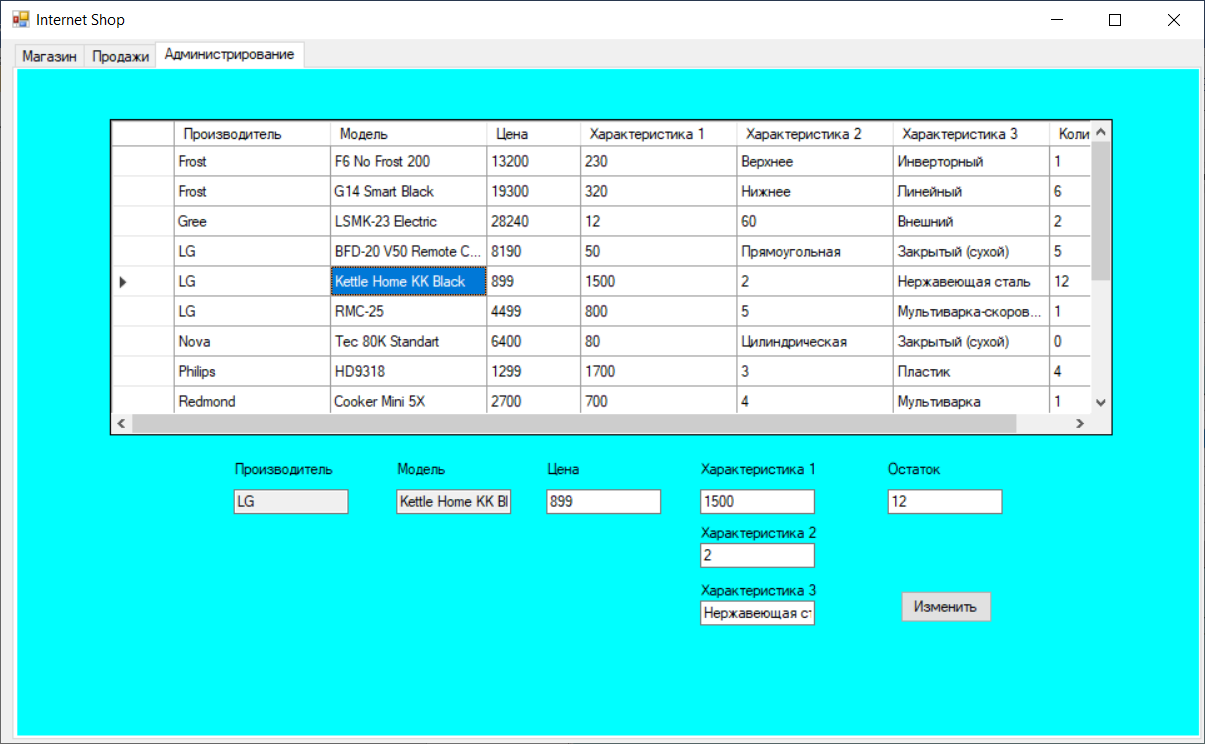


Рисунок 4‑3 – Налаштування характеристик

# ВИСНОВКИ

У даній роботі спроектували та реалізували базу даних та прикладну програму графічного інтерфейсу користувача для магазину побутової техніки.

У програмі додали можливості пошуку інформації, додавання, видалення та змінення даних.

Реалізація програми здійснена з використанням об’єктно-орієнтованих технологій. Представили необхідну інформацію про програму та способи її застосування.

Актуальність даної програми полягає в тому, що в багатьох комерційних системах потрібно зберігати велику кількість інформації, що стосується товарів, клієнтів, продажів і таке інше. Таку інформацію зручно зберігати розподілено, у різних таблицях, але у вигляді однієї загальної бази даних.

Виконали наступні пункти:

1. Створення бази даних
2. Створення необхідних тригерів і представлень
3. Підключення бази даних до програми
4. Створення необхідного інтерфейсу, що дозволить взаємодіяти з базою даних
5. Тестування програми за допомогою початкових даних, заздалегідь внесених до бази даних

Після виконання цих дій, розробили технічну документацію до програми, в якій роз’яснили способи та особливості використання основних можливостей програми.

# СПИСОК ЛІТЕРАТУРИ

1. Вивчаємо SQL. 3 видання – Алан Болье – «Діалектика», 2021. — С. 402.
2. Майк Гандерлой, Джозеф Джорден, Дейвид Чанц. Освоєння Microsoft SQL Server 2005 – Освоение Microsoft SQL Server 2005. – М. : «Діалектика», 2007. – С. 2204.
3. Організація баз даних: практичний курс: Навч. посіб. для студ. / А. Ю. Берко, О. М. Верес; Нац. ун-т «Львів. політехніка». – Л., 2003. –149 c.
4. С.Д. Кузнецов. Три манифеста баз данных: ретроспектива и перспективы. CIT Forum. [Архів](https://www.webcitation.org/65IoWtWrN?url=http://citforum.ru/database/articles/manifests/) оригіналу за 8 лютого 2012. Процитовано 12 червня 2011.
5. Рябцев В. В. – Навчальна програма дисципліни «Теорія баз даних». – К. : МАУП, 2018. – 22 с.
6. Сахіл Малік Microsoft ADO.NET 2.0 для професіоналів – Pro ADO.NET 2.0. – М.: «Вільямс», 2006. — С. 560.
7. Язык программирования C# 9 и платформа .NET 5: основные  
   принципы и практики программирования, том 2, 10-е изд./Эндрю  
   Троелсен, Филипп Джепикс; пер. с англ. Ю.Н. Артеменко. — Киев. :  
   “Диалектика”, 2022. — 632 с.: ил. — Парал. тит. англ.
8. Коноваленко І. В. – Програмування мовою C#6.0. – ТНТУ, Тернопіль. – 2016. – 227 с.

# ДОДАТОК

use master

go

if exists(select \* from sys.databases where name = 'coursework\_shop')

begin

alter database coursework\_shop set single\_user with rollback immediate

drop database coursework\_shop

end

go

create database coursework\_shop

go

use coursework\_shop

go

create table product

(

id int not null identity(1,1) primary key,

price int not null default 0

)

go

insert into product(price) values

(8400),

(899),

(13200),

(5700),

(2700),

(4499),

(8190),

(12000),

(19300),

(28240),

(12300),

(1299),

(4200),

(15800),

(6400)

go

create table storage

(

id int not null identity(1,1) primary key,

id\_product int not null foreign key references product(id),

k int not null default 0

)

go

insert into storage(id\_product, k) values

(1, 4),

(2, 12),

(3, 1),

(4, 0),

(5, 1),

(6, 1),

(7, 5),

(8, 1),

(9, 6),

(10, 2),

(11, 1),

(12, 4),

(13, 5),

(14, 1),

(15, 0)

go

create table sale

(

id int not null identity(1,1) primary key,

date\_sale date,

sum\_sale int not null default 0

)

go

create trigger tr1

on sale

instead of delete

as

begin

declare @id int = (select top 1 id from deleted)

delete from sale\_row where id\_sale = @id

delete from sale where id = @id

end

go

insert into sale(date\_sale, sum\_sale) values

('2022-09-22', 12000),

('2022-09-30', 3940),

('2022-10-02', 11400),

('2022-11-21', 899),

('2022-11-21', 8400),

('2022-11-24', 18700),

('2022-11-25', 4499)

go

create table sale\_row

(

id int not null identity(1,1) primary key,

id\_sale int not null foreign key references sale(id),

id\_product int not null foreign key references product(id),

k int not null default 1

)

go

create trigger tr2

on sale\_row

after update, insert

as

begin

declare @id int = (select top 1 id\_sale from inserted)

update sale set sum\_sale = (select sum(product.price \* sale\_row.k) from sale\_row join product on sale\_row.id\_product = product.id where id\_sale = @id) where id = @id

end

go

insert into sale\_row(id\_sale, id\_product, k) values (1, 8, 1)

go

insert into sale\_row(id\_sale, id\_product, k) values (2, 5, 1)

go

insert into sale\_row(id\_sale, id\_product, k) values (2, 10, 1)

go

insert into sale\_row(id\_sale, id\_product, k) values (3, 4, 2)

go

insert into sale\_row(id\_sale, id\_product, k) values (4, 2, 1)

go

insert into sale\_row(id\_sale, id\_product, k) values (5, 13, 2)

go

insert into sale\_row(id\_sale, id\_product, k) values (6, 15, 1)

go

insert into sale\_row(id\_sale, id\_product, k) values (6, 11, 1)

go

insert into sale\_row(id\_sale, id\_product, k) values (7, 6, 1)

go

create table group\_product

(

id int not null identity(1,1) primary key,

obj\_name nvarchar(50) not null,

group\_name nvarchar(50) not null

)

go

insert into group\_product(obj\_name, group\_name) values

(N'Холодильник', N'Крупная бытовая техника'),

(N'Стиральная машина', N'Крупная бытовая техника'),

(N'Бойлер', N'Климатическая техника'),

(N'Кондиционер', N'Климатическая техника'),

(N'Электрочайник', N'Техника для кухни'),

(N'Мультиварка', N'Техника для кухни')

go

create table info

(

id int not null identity(1,1) primary key,

id\_product int not null foreign key references product(id),

brand nvarchar(50) not null,

model nvarchar(50) not null,

country nvarchar(50)

)

go

insert into info(id\_product, brand, model, country) values

(1, N'Samsung', N'MF-230 V80', N'Корея'),

(2, N'LG', N'Kettle Home KK Black', N'Германия'),

(3, N'Frost', N'F6 No Frost 200', N'Польша'),

(4, N'Samsung', N'C10 Remote White', N'Китай'),

(5, N'Redmond', N'Cooker Mini 5X', N'Германия'),

(6, N'LG', N'RMC-25', N'Польша'),

(7, N'LG', N'BFD-20 V50 Remote Control', N'Германия'),

(8, N'Samsung', N'Wash LM-340 White', N'Корея'),

(9, N'Frost', N'G14 Smart Black', N'Польша'),

(10, N'Gree', N'LSMK-23 Electric', N'Франция'),

(11, N'Saturn', N'CW 286', N'Италия'),

(12, N'Philips', N'HD9318', N'Корея'),

(13, N'Redmond', N'RMC-507 Auto', N'Франция'),

(14, N'Samsung', N'OMTA46F', N'Германия'),

(15, N'Nova', N'Tec 80K Standart', N'Германия')

go

create table fridge

(

id int not null identity(1,1) primary key,

id\_group int default 1 foreign key references group\_product(id),

id\_product int not null foreign key references info(id),

volume int,

freezer nvarchar(50),

compressor nvarchar(50)

)

go

insert into fridge(id\_product, volume, freezer, compressor) values

(3, 230, N'Верхнее', N'Инверторный'),

(9, 320, N'Нижнее', N'Линейный'),

(11, 350, N'Нижнее', N'Линейный')

go

create table washing\_machine

(

id int not null identity(1,1) primary key,

id\_group int default 2 foreign key references group\_product(id),

id\_product int not null foreign key references info(id),

max\_weight int,

engine nvarchar(50),

loading nvarchar(50)

)

go

insert into washing\_machine(id\_product, max\_weight, engine, loading) values

(8, 8, N'Инверторный', N'Фронтальная'),

(14, 9, N'Коллекторный', N'Вертикальная')

go

create table boiler

(

id int not null identity(1,1) primary key,

id\_group int default 3 foreign key references group\_product(id),

id\_product int not null foreign key references info(id),

volume int,

shape nvarchar(50),

heater nvarchar(50)

)

go

insert into boiler(id\_product, volume, shape, heater) values

(1, 80, N'Цилиндрическая', N'Открытый (мокрый)'),

(7, 50, N'Прямоугольная', N'Закрытый (сухой)'),

(15, 80, N'Цилиндрическая', N'Закрытый (сухой)')

go

create table conditioner

(

id int not null identity(1,1) primary key,

id\_group int default 4 foreign key references group\_product(id),

id\_product int not null foreign key references info(id),

mark int,

room\_square int,

kind nvarchar(50)

)

go

insert into conditioner(id\_product, mark, room\_square, kind) values

(4, 9, 25, N'Сплит-система'),

(10, 12, 60, N'Внешний')

go

create table electric\_kettle

(

id int not null identity(1,1) primary key,

id\_group int default 5 foreign key references group\_product(id),

id\_product int not null foreign key references info(id),

voltage\_power int,

volume int,

material nvarchar(50)

)

go

insert into electric\_kettle(id\_product, voltage\_power, volume, material) values

(2, 1500, 2, N'Нержавеющая сталь'),

(12, 1700, 3, N'Пластик')

go

create table multicooker

(

id int not null identity(1,1) primary key,

id\_group int default 6 foreign key references group\_product(id),

id\_product int not null foreign key references info(id),

voltage\_power int,

volume int,

kind nvarchar(50)

)

go

insert into multicooker(id\_product, voltage\_power, volume, kind) values

(5, 700, 4, N'Мультиварка'),

(6, 800, 5, N'Мультиварка-скороварка'),

(13, 1000, 5, N'Рисоварка')

go

create view view\_client

as

select group\_product.obj\_name as 'Тип', info.brand as 'Производитель', info.model as 'Модель', product.price as 'Цена'

from (product join info on info.id\_product = product.id join fridge on fridge.id\_product = info.id join group\_product on fridge.id\_group = group\_product.id)

union

select group\_product.obj\_name as 'Тип', info.brand as 'Производитель', info.model as 'Модель', product.price as 'Цена'

from (product join info on info.id\_product = product.id join washing\_machine on washing\_machine.id\_product = info.id join group\_product on washing\_machine.id\_group = group\_product.id)

union

select group\_product.obj\_name as 'Тип', info.brand as 'Производитель', info.model as 'Модель', product.price as 'Цена'

from (product join info on info.id\_product = product.id join boiler on boiler.id\_product = info.id join group\_product on boiler.id\_group = group\_product.id)

union

select group\_product.obj\_name as 'Тип', info.brand as 'Производитель', info.model as 'Модель', product.price as 'Цена'

from (product join info on info.id\_product = product.id join conditioner on conditioner.id\_product = info.id join group\_product on conditioner.id\_group = group\_product.id)

union

select group\_product.obj\_name as 'Тип', info.brand as 'Производитель', info.model as 'Модель', product.price as 'Цена'

from (product join info on info.id\_product = product.id join electric\_kettle on electric\_kettle.id\_product = info.id join group\_product on electric\_kettle.id\_group = group\_product.id)

union

select group\_product.obj\_name as 'Тип', info.brand as 'Производитель', info.model as 'Модель', product.price as 'Цена'

from (product join info on info.id\_product = product.id join multicooker on multicooker.id\_product = info.id join group\_product on multicooker.id\_group = group\_product.id)

go

create view view\_sales

as

select sale.id as 'Номер чека', sale.date\_sale as 'Дата', info.brand as 'Производитель', info.model as 'Модель', sale\_row.k as 'Количество', product.price as 'Цена'

from sale join sale\_row on sale\_row.id\_sale = sale.id join product on sale\_row.id\_product = product.id join info on info.id\_product = product.id

go

create view view\_admin

as

select info.brand as 'Производитель', info.model as 'Модель', product.price as 'Цена', CAST(conditioner.mark as varchar(50)) as 'Характеристика 1', CAST(conditioner.room\_square as varchar(50)) as 'Характеристика 2', conditioner.kind as 'Характеристика 3', storage.k as 'Количество'

from (product join info on info.id\_product = product.id join conditioner on conditioner.id\_product = info.id join storage on storage.id\_product = product.id)

union

select info.brand as 'Производитель', info.model as 'Модель', product.price as 'Цена', CAST(electric\_kettle.voltage\_power as varchar(50)) as 'Характеристика 1', CAST(electric\_kettle.volume as varchar(50)) as 'Характеристика 2', electric\_kettle.material as 'Характеристика 3', storage.k as 'Количество'

from (product join info on info.id\_product = product.id join electric\_kettle on electric\_kettle.id\_product = info.id join storage on storage.id\_product = product.id)

union

select info.brand as 'Производитель', info.model as 'Модель', product.price as 'Цена', CAST(multicooker.voltage\_power as varchar(50)) as 'Характеристика 1', CAST(multicooker.volume as varchar(50)) as 'Характеристика 2', multicooker.kind as 'Характеристика 3', storage.k as 'Количество'

from (product join info on info.id\_product = product.id join multicooker on multicooker.id\_product = info.id join storage on storage.id\_product = product.id)

union

select info.brand as 'Производитель', info.model as 'Модель', product.price as 'Цена', fridge.volume as 'Характеристика 1', fridge.freezer as 'Характеристика 2', fridge.compressor as 'Характеристика 3', storage.k as 'Количество'

from (product join info on info.id\_product = product.id join fridge on fridge.id\_product = info.id join storage on storage.id\_product = product.id)

union

select info.brand as 'Производитель', info.model as 'Модель', product.price as 'Цена', washing\_machine.max\_weight as 'Характеристика 1', washing\_machine.engine as 'Характеристика 2', washing\_machine.loading as 'Характеристика 3', storage.k as 'Количество'

from (product join info on info.id\_product = product.id join washing\_machine on washing\_machine.id\_product = info.id join storage on storage.id\_product = product.id)

union

select info.brand as 'Производитель', info.model as 'Модель', product.price as 'Цена', boiler.volume as 'Характеристика 1', boiler.shape as 'Характеристика 2', boiler.heater as 'Характеристика 3', storage.k as 'Количество'

from (product join info on info.id\_product = product.id join boiler on boiler.id\_product = info.id join storage on storage.id\_product = product.id)

go

alter database coursework\_shop set multi\_user

go

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace shop

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "coursework\_shopDataSet.view\_admin". При необходимости она может быть перемещена или удалена.

this.view\_adminTableAdapter.Fill(this.coursework\_shopDataSet.view\_admin);

// TODO: данная строка кода позволяет загрузить данные в таблицу "coursework\_shopDataSet.view\_sales". При необходимости она может быть перемещена или удалена.

// TODO: данная строка кода позволяет загрузить данные в таблицу "coursework\_shopDataSet.view\_client". При необходимости она может быть перемещена или удалена.

this.view\_clientTableAdapter.Fill(this.coursework\_shopDataSet.view\_client);

// TODO: данная строка кода позволяет загрузить данные в таблицу "coursework\_shopDataSet.product". При необходимости она может быть перемещена или удалена.

this.productTableAdapter.Fill(this.coursework\_shopDataSet.product);

// TODO: данная строка кода позволяет загрузить данные в таблицу "coursework\_shopDataSet.group\_product". При необходимости она может быть перемещена или удалена.

this.group\_productTableAdapter.Fill(this.coursework\_shopDataSet.group\_product);

this.view\_salesTableAdapter.Fill(this.coursework\_shopDataSet.view\_sales);

foreach (DataRow row in this.coursework\_shopDataSet.group\_product.Rows)

{

checkedListBox1.Items.Add(row["obj\_name"].ToString());

}

string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\coursework\_shop.mdf;Integrated Security=True;Connect Timeout=30";

string sql = "select Производитель from view\_client group by Производитель";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

// Создаем объект DataAdapter

SqlDataAdapter adapter = new SqlDataAdapter(sql, connection);

// Создаем объект Dataset

DataSet ds = new DataSet();

// Заполняем Dataset

adapter.Fill(ds);

// Отображаем данные

foreach (DataRow row in ds.Tables[0].Rows)

checkedListBox2.Items.Add(row["Производитель"].ToString());

}

}

private void tabPage1\_Click(object sender, EventArgs e)

{

}

private void panel1\_Paint(object sender, PaintEventArgs e)

{

}

private void groupBox1\_Enter(object sender, EventArgs e)

{

}

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

{

try

{

this.group\_productTableAdapter.FillBy(this.coursework\_shopDataSet.group\_product);

}

catch (System.Exception ex)

{

System.Windows.Forms.MessageBox.Show(ex.Message);

}

}

private void checkedListBox1\_ItemCheck(object sender, ItemCheckEventArgs e)

{

string sql = "1 = 1";

List<string> checkedItems1 = new List<string>();

foreach (var item in checkedListBox1.CheckedItems)

checkedItems1.Add(item.ToString());

if (e.NewValue == CheckState.Checked)

checkedItems1.Add(checkedListBox1.Items[e.Index].ToString());

else

checkedItems1.Remove(checkedListBox1.Items[e.Index].ToString());

List<string> checkedItems2 = new List<string>();

foreach (var item in checkedListBox2.CheckedItems)

checkedItems2.Add(item.ToString());

if (checkedItems1.Count == 0 && checkedItems2.Count == 0)

{

richTextBox1.Text = "Фильтрация: нет";

}

else

{

richTextBox1.Text = "Фильтрация:";

}

if (checkedItems1.Count > 0)

{

sql += " and (1 = 0";

foreach (string item in checkedItems1)

{

sql += $" or Тип = '{item}'";

richTextBox1.Text += $" {item};";

}

sql += ")";

}

if (checkedItems2.Count > 0)

{

sql += " and (1 = 0";

foreach (string item in checkedItems2)

{

sql += $" or Производитель = '{item}'";

richTextBox1.Text += $" {item};";

}

sql += ")";

}

if (textBox1.Text.Length > 0)

{

sql += " and";

sql += $" Цена >= {textBox1.Text}";

richTextBox1.Text += $" Цена от {textBox1.Text};";

}

if (textBox2.Text.Length > 0)

{

sql += " and";

sql += $" Цена <= {textBox2.Text}";

richTextBox1.Text += $" Цена до {textBox2.Text};";

}

viewclientBindingSource.Filter = sql;

}

private void checkedListBox2\_ItemCheck(object sender, ItemCheckEventArgs e)

{

string sql = "1 = 1";

List<string> checkedItems1 = new List<string>();

foreach (var item in checkedListBox1.CheckedItems)

checkedItems1.Add(item.ToString());

List<string> checkedItems2 = new List<string>();

foreach (var item in checkedListBox2.CheckedItems)

checkedItems2.Add(item.ToString());

if (e.NewValue == CheckState.Checked)

checkedItems2.Add(checkedListBox2.Items[e.Index].ToString());

else

checkedItems2.Remove(checkedListBox2.Items[e.Index].ToString());

if (checkedItems1.Count == 0 && checkedItems2.Count == 0)

{

richTextBox1.Text = "Фильтрация: нет";

}

else

{

richTextBox1.Text = "Фильтрация:";

}

if (checkedItems1.Count > 0)

{

sql += " and (1 = 0";

foreach (string item in checkedItems1)

{

sql += $" or Тип = '{item}'";

richTextBox1.Text += $" {item};";

}

sql += ")";

}

if (checkedItems2.Count > 0)

{

sql += " and (1 = 0";

foreach (string item in checkedItems2)

{

sql += $" or Производитель = '{item}'";

richTextBox1.Text += $" {item};";

}

sql += ")";

}

if (textBox1.Text.Length > 0)

{

sql += " and";

sql += $" Цена >= {textBox1.Text}";

richTextBox1.Text += $" Цена от {textBox1.Text};";

}

if (textBox2.Text.Length > 0)

{

sql += " and";

sql += $" Цена <= {textBox2.Text}";

richTextBox1.Text += $" Цена до {textBox2.Text};";

}

viewclientBindingSource.Filter = sql;

}

private void button1\_Click(object sender, EventArgs e)

{

for (int i = 0; i < checkedListBox1.Items.Count; i++)

{

checkedListBox1.SetItemChecked(i, false);

}

for (int i = 0; i < checkedListBox2.Items.Count; i++)

{

checkedListBox2.SetItemChecked(i, false);

}

textBox1.Text = "";

textBox2.Text = "";

richTextBox1.Text = "Фильтрация: нет";

viewclientBindingSource.Filter = null;

}

private void label2\_Click(object sender, EventArgs e)

{

}

private void button2\_Click(object sender, EventArgs e)

{

string sql = "1 = 1";

List<string> checkedItems1 = new List<string>();

foreach (var item in checkedListBox1.CheckedItems)

checkedItems1.Add(item.ToString());

List<string> checkedItems2 = new List<string>();

foreach (var item in checkedListBox2.CheckedItems)

checkedItems2.Add(item.ToString());

if (checkedItems1.Count == 0 && checkedItems2.Count == 0)

{

richTextBox1.Text = "Фильтрация: нет";

}

else

{

richTextBox1.Text = "Фильтрация:";

}

if (checkedItems1.Count > 0)

{

sql += " and (1 = 0";

foreach (string item in checkedItems1)

{

sql += $" or Тип = '{item}'";

richTextBox1.Text += $" {item};";

}

sql += ")";

}

if (checkedItems2.Count > 0)

{

sql += " and (1 = 0";

foreach (string item in checkedItems2)

{

sql += $" or Производитель = '{item}'";

richTextBox1.Text += $" {item};";

}

sql += ")";

}

if (textBox1.Text.Length > 0)

{

sql += " and";

sql += $" Цена >= {textBox1.Text}";

richTextBox1.Text += $" Цена от {textBox1.Text};";

}

if (textBox2.Text.Length > 0)

{

sql += " and";

sql += $" Цена <= {textBox2.Text}";

richTextBox1.Text += $" Цена до {textBox2.Text};";

}

viewclientBindingSource.Filter = sql;

}

private void label8\_Click(object sender, EventArgs e)

{

}

private void dataGridView1\_CellClick(object sender, DataGridViewCellEventArgs e)

{

string cons = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\coursework\_shop.mdf;Integrated Security=True;Connect Timeout=30";

SqlConnection con = new SqlConnection(cons);

con.Open();

string q;

string type = dataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString();

string pr = dataGridView1.Rows[e.RowIndex].Cells[1].Value.ToString();

string mo = dataGridView1.Rows[e.RowIndex].Cells[2].Value.ToString();

if (type == "Холодильник")

{

q = @"select product.price as 'Цена', info.brand as 'Производитель', info.model as 'Модель', info.country as 'Страна', fridge.volume as 'Характеристика 1', fridge.freezer as 'Характеристика 2', fridge.compressor as 'Характеристика 3'

from (product join info on info.id\_product = product.id join fridge on fridge.id\_product = info.id)

where info.brand = N'" + pr + "' and info.model = N'" + mo + "'";

}

else if (type == "Стиральная машина")

{

q = @"select product.price as 'Цена', info.brand as 'Производитель', info.model as 'Модель', info.country as 'Страна', washing\_machine.max\_weight as 'Характеристика 1', washing\_machine.engine as 'Характеристика 2', washing\_machine.loading as 'Характеристика 3'

from (product join info on info.id\_product = product.id join washing\_machine on washing\_machine.id\_product = info.id)

where info.brand = N'" + pr + "' and info.model = N'" + mo + "'";

}

else if (type == "Бойлер")

{

q = @"select product.price as 'Цена', info.brand as 'Производитель', info.model as 'Модель', info.country as 'Страна', boiler.volume as 'Характеристика 1', boiler.shape as 'Характеристика 2', boiler.heater as 'Характеристика 3'

from (product join info on info.id\_product = product.id join boiler on boiler.id\_product = info.id)

where info.brand = N'" + pr + "' and info.model = N'" + mo + "'";

}

else if (type == "Кондиционер")

{

q = @"select product.price as 'Цена', info.brand as 'Производитель', info.model as 'Модель', info.country as 'Страна', conditioner.mark as 'Характеристика 1', conditioner.room\_square as 'Характеристика 2', conditioner.kind as 'Характеристика 3'

from (product join info on info.id\_product = product.id join conditioner on conditioner.id\_product = info.id)

where info.brand = N'" + pr + "' and info.model = N'" + mo + "'";

}

else if (type == "Электрочайник")

{

q = @"select product.price as 'Цена', info.brand as 'Производитель', info.model as 'Модель', info.country as 'Страна', electric\_kettle.voltage\_power as 'Характеристика 1', electric\_kettle.volume as 'Характеристика 2', electric\_kettle.material as 'Характеристика 3'

from (product join info on info.id\_product = product.id join electric\_kettle on electric\_kettle.id\_product = info.id)

where info.brand = N'" + pr + "' and info.model = N'" + mo + "'";

}

else if (type == "Мультиварка")

{

q = @"select product.price as 'Цена', info.brand as 'Производитель', info.model as 'Модель', info.country as 'Страна', multicooker.voltage\_power as 'Характеристика 1', multicooker.volume as 'Характеристика 2', multicooker.kind as 'Характеристика 3'

from (product join info on info.id\_product = product.id join multicooker on multicooker.id\_product = info.id)

where info.brand = N'" + pr + "' and info.model = N'" + mo + "'";

}

else return;

SqlCommand cmd = new SqlCommand(q, con);

SqlDataReader dr = cmd.ExecuteReader();

while (dr.Read())

{

label3.Text = $"Цена: {dr.GetValue(0)} грн";

label4.Text = $"Производитель: {dr.GetValue(1)}";

label5.Text = $"Модель: {dr.GetValue(2)}";

label6.Text = $"Страна: {dr.GetValue(3)}";

if (type == "Холодильник")

{

label7.Text = $"Объем холодильной камеры: {dr.GetValue(4)} л";

label8.Text = $"Расположение морозильного отдела: {dr.GetValue(5)}";

label9.Text = $"Тип компрессора: {dr.GetValue(6)}";

}

else if (type == "Стиральная машина")

{

label7.Text = $"Максимальный вес загрузки: {dr.GetValue(4)} кг";

label8.Text = $"Тип двигателя: {dr.GetValue(5)}";

label9.Text = $"Тип загрузки: {dr.GetValue(6)}";

}

else if (type == "Бойлер")

{

label7.Text = $"Объем: {dr.GetValue(4)} л";

label8.Text = $"Форма корпуса: {dr.GetValue(5)}";

label9.Text = $"Тип ТЭНа: {dr.GetValue(6)}";

}

else if (type == "Кондиционер")

{

label7.Text = $"Маркировка: {dr.GetValue(4)}";

label8.Text = $"Обслуживаемая площадь: {dr.GetValue(5)} м^2";

label9.Text = $"Тип: {dr.GetValue(6)}";

}

else if (type == "Электрочайник")

{

label7.Text = $"Мощность: {dr.GetValue(4)} Вт";

label8.Text = $"Объем: {dr.GetValue(5)} л";

label9.Text = $"Материал корпуса: {dr.GetValue(6)}";

}

else if (type == "Мультиварка")

{

label7.Text = $"Мощность: {dr.GetValue(4)} Вт";

label8.Text = $"Объем: {dr.GetValue(5)} л";

label9.Text = $"Тип: {dr.GetValue(6)}";

}

}

dr.Close();

con.Close();

}

private void button4\_Click(object sender, EventArgs e)

{

}

private void tabPage2\_Click(object sender, EventArgs e)

{

}

private void button3\_Click(object sender, EventArgs e)

{

string sql = "1 = 1";

if (textBox3.Text.Length > 0)

sql += $" and [Номер чека] = '{textBox3.Text}'";

if (textBox4.Text.Length > 0)

sql += $" and Производитель = '{textBox4.Text}'";

if (textBox5.Text.Length > 0)

sql += $" and Модель = '{textBox5.Text}'";

if (textBox6.Text.Length > 0)

sql += $" and Дата = '{textBox6.Text}'";

viewsalesBindingSource.Filter = sql;

}

private void button6\_Click(object sender, EventArgs e)

{

textBox3.Text = "";

textBox4.Text = "";

textBox5.Text = "";

textBox6.Text = "";

viewsalesBindingSource.Filter = null;

}

private void button4\_Click\_1(object sender, EventArgs e)

{

panel3.Visible = true;

}

private void button5\_Click(object sender, EventArgs e)

{

if (textBox7.Text.Length > 0 && textBox8.Text.Length > 0 && textBox9.Text.Length > 0)

{

string cons = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\coursework\_shop.mdf;Integrated Security=True;Connect Timeout=30";

SqlConnection con = new SqlConnection(cons);

con.Open();

System.DateTime date = System.DateTime.Now;

string s = $"insert into sale(date\_sale) values ('{date.Year}-{date.Month}-{date.Day}')";

SqlCommand cmd = new SqlCommand(s, con);

int k = cmd.ExecuteNonQuery();

int id\_sale = 0;

s = @"select top 1 id from sale order by id desc";

cmd = new SqlCommand(s, con);

SqlDataReader dr = cmd.ExecuteReader();

while(dr.Read())

{

id\_sale = Int32.Parse(dr.GetValue(0).ToString());

}

dr.Close();

int id\_prod = 0;

s = $"select top 1 product.id from product join info on info.id\_product = product.id where brand = N'{textBox7.Text}' and model = N'{textBox8.Text}'";

cmd = new SqlCommand(s, con);

dr = cmd.ExecuteReader();

while (dr.Read())

{

id\_prod = Int32.Parse(dr.GetValue(0).ToString());

}

dr.Close();

s = $"insert into sale\_row(id\_sale, id\_product, k) values ({id\_sale}, {id\_prod}, {Int32.Parse(textBox9.Text)})";

cmd = new SqlCommand(s, con);

k = cmd.ExecuteNonQuery();

this.view\_salesTableAdapter.Fill(this.coursework\_shopDataSet.view\_sales);

con.Close();

}

textBox7.Text = "";

textBox8.Text = "";

textBox9.Text = "";

panel3.Visible = false;

}

private void dataGridView3\_CellClick(object sender, DataGridViewCellEventArgs e)

{

int row = e.RowIndex;

textBox10.Text = dataGridView3.Rows[row].Cells[0].Value.ToString();

textBox11.Text = dataGridView3.Rows[row].Cells[1].Value.ToString();

textBox12.Text = dataGridView3.Rows[row].Cells[2].Value.ToString();

textBox13.Text = dataGridView3.Rows[row].Cells[3].Value.ToString();

textBox14.Text = dataGridView3.Rows[row].Cells[4].Value.ToString();

textBox15.Text = dataGridView3.Rows[row].Cells[5].Value.ToString();

textBox16.Text = dataGridView3.Rows[row].Cells[6].Value.ToString();

}

private void button7\_Click(object sender, EventArgs e)

{

string cons = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\coursework\_shop.mdf;Integrated Security=True;Connect Timeout=30";

SqlConnection con = new SqlConnection(cons);

con.Open();

string s = $"select top 1 Тип from view\_client where Производитель = N'{textBox10.Text}' and Модель = N'{textBox11.Text}'";

SqlCommand cmd = new SqlCommand(s, con);

SqlDataReader dr = cmd.ExecuteReader();

string type = "";

while (dr.Read())

{

type = dr.GetValue(0).ToString();

}

dr.Close();

s = $"select top 1 product.id from product join info on info.id\_product = product.id where info.brand = N'{textBox10.Text}' and info.model = N'{textBox11.Text}'";

cmd = new SqlCommand(s, con);

dr = cmd.ExecuteReader();

int id = 0;

while (dr.Read())

{

id = Int32.Parse(dr.GetValue(0).ToString());

}

dr.Close();

s = $"update product set price = {Int32.Parse(textBox12.Text)} where id = {id}";

cmd = new SqlCommand(s, con);

int k = cmd.ExecuteNonQuery();

s = $"update storage set k = {Int32.Parse(textBox16.Text)} where id\_product = {id}";

cmd = new SqlCommand(s, con);

k = cmd.ExecuteNonQuery();

if (type == "Холодильник")

{

s = $"update fridge set volume = {Int32.Parse(textBox13.Text)}, freezer = N'{textBox14.Text}', compressor = N'{textBox15.Text}' where id\_product = {id}";

cmd = new SqlCommand(s, con);

k = cmd.ExecuteNonQuery();

}

else if (type == "Стиральная машина")

{

s = $"update washing\_machine set max\_weight = {Int32.Parse(textBox13.Text)}, engine = N'{textBox14.Text}', loading = N'{textBox15.Text}' where id\_product = {id}";

cmd = new SqlCommand(s, con);

k = cmd.ExecuteNonQuery();

}

else if (type == "Бойлер")

{

s = $"update boiler set volume = {Int32.Parse(textBox13.Text)}, shape = N'{textBox14.Text}', heater = N'{textBox15.Text}' where id\_product = {id}";

cmd = new SqlCommand(s, con);

k = cmd.ExecuteNonQuery();

}

else if (type == "Кондиционер")

{

s = $"update conditioner set mark = {Int32.Parse(textBox13.Text)}, room\_square = {Int32.Parse(textBox14.Text)}, kind = N'{textBox15.Text}' where id\_product = {id}";

cmd = new SqlCommand(s, con);

k = cmd.ExecuteNonQuery();

}

else if (type == "Электрочайник")

{

s = $"update electric\_kettle set voltage\_power = {Int32.Parse(textBox13.Text)}, volume = {Int32.Parse(textBox14.Text)}, material = N'{textBox15.Text}' where id\_product = {id}";

cmd = new SqlCommand(s, con);

k = cmd.ExecuteNonQuery();

}

else if (type == "Мультиварка")

{

s = $"update multicooker set voltage\_power = {Int32.Parse(textBox13.Text)}, volume = {Int32.Parse(textBox14.Text)}, kind = N'{textBox15.Text}' where id\_product = {id}";

cmd = new SqlCommand(s, con);

k = cmd.ExecuteNonQuery();

}

this.view\_clientTableAdapter.Fill(this.coursework\_shopDataSet.view\_client);

this.view\_salesTableAdapter.Fill(this.coursework\_shopDataSet.view\_sales);

this.view\_adminTableAdapter.Fill(this.coursework\_shopDataSet.view\_admin);

con.Close();

}

private void button8\_Click(object sender, EventArgs e)

{

string cons = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\coursework\_shop.mdf;Integrated Security=True;Connect Timeout=30";

SqlConnection con = new SqlConnection(cons);

con.Open();

int id = Int32.Parse(textBox17.Text);

string s = $"delete from sale where id = {id}";

SqlCommand cmd = new SqlCommand(s, con);

int k = cmd.ExecuteNonQuery();

this.view\_salesTableAdapter.Fill(this.coursework\_shopDataSet.view\_sales);

con.Close();

textBox17.Text = "";

}

private void dataGridView2\_CellClick(object sender, DataGridViewCellEventArgs e)

{

textBox17.Text = dataGridView2.Rows[e.RowIndex].Cells[0].Value.ToString();

}

}

}