МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ДНІПРОВСЬКА ПОЛІТЕХНІКА»

Факультет: Інформаційних технологій

Кафедра: Програмного забезпечення комп’ютерних систем

(повна назва кафедри)

**КУРСОВА РОБОТА**

з дисципліни: **ОРГАНІЗАЦІЯ БАЗ ДАНИХ І БАЗ ЗНАНЬ**

(назва дисципліни)

на тему:\_ «**Розробка автоматизованої інформаційної системи для постачальника газу»**

Студента 2 курсу, групи 122-20-2

спеціальності 122 Комп’ютерні науки

\_\_Півень І.С.

(прізвище та ініціали)

Керівник \_доцент, к.т.н., доцент \_\_\_\_\_\_

\_\_\_Кабак Л.В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_Оцінка: ECTS \_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кабак Л.В.

(підпис) (прізвище та ініціали)

м. Дніпро - 2022 рік

# 1. ЗМІСТ

[**1. ЗМІСТ** 2](#_Toc104835217)

[2. ПОСТАНОВКА ЗАДАЧІ 3](#_Toc104835218)

[3. СТВОРЕННЯ ЛОГІЧНОЇ МОДЕЛІ 4](#_Toc104835219)

[3.1. Визначення об'єктів 4](#_Toc104835220)

[3.2. Визначення взаємозв'язків між сутностями 5](#_Toc104835221)

[3.3. Завдання первинних і альтернативних ключів, визначення атрибутів сутностей 6](#_Toc104835222)

[4. СТВОРЕННЯ ФІЗИЧНОЇ МОДЕЛІ 8](#_Toc104835223)

[5. ОПИС ПРОГРАМИ 10](#_Toc104835224)

[5.1. Логічна структура 10](#_Toc104835225)

[5.2. Екранні форми 11](#_Toc104835226)

[Форма «Customers» 11](#_Toc104835227)

[Форма «Counters» 17](#_Toc104835228)

[5.3. Звіти 22](#_Toc104835229)

[Звіт «Customers Report» 22](#_Toc104835230)

[Звіт «Counters Report» 22](#_Toc104835231)

[5.4. Текст програми 24](#_Toc104835232)

[5.5. Список літератури 27](#_Toc104835233)

2. ПОСТАНОВКА ЗАДАЧІ

*Мета курсової роботи* - освоєння технології проектування і створення автоматизованих інформаційних систем (АІС), у середовищі швидкої розробки додатків (Rapid Application Development - RAD) MS Visual Studio C#, з використанням мови структурних запитів SQL.

*Задачі проекту* – створення концептуальної моделі конкретної предметної області, логічної моделі даних і, нарешті, фізичної моделі даних, на підставі якої створюється розподілена база даних, для роботи з якою необхідно створити додаток засобами RAD-системи MS Visual Studio C#. База даних розробляється за допомогою мови структурних запитів SQL. Тригери, процедури та функції бази даних розробляються за допомогою програмної мови PL/SQL.

*Завдання проекту* - створення інформаційної системи з використанням бази даних для місцевого підприємства, яке займається постачанням газу для населення, виконує розрахунки з клієнтами.

Дані мають наступну структуру: ім’я та прізвище, домашня адреса (назва вулиці, номер дома), місто і район, початкові показники лічильнику газу, кінцеві показники, початкова дата, кінцева дата, номер лічильника.

Розроблений додаток повинен створювати звіти за заданий період.

3. СТВОРЕННЯ ЛОГІЧНОЇ МОДЕЛІ

3.1. Визначення об'єктів

Виходячи з поставленої задачі, виділимо наступні сутності:

1. КЛІЄНТ;
2. ЛІЧИЛЬНИК.

Об'єкти є поняттями реального світу. У реляційній моделі даних вони моделюються таблицями. Стовпці цих таблиць мають в собі властивості об’єктів, що називаються атрибутами.

Згідно поставленого завдання, об’єкт КЛІЄНТ має наступні атрибути:

* ім’я клієнта;
* прізвище клієнта;
* назва вулиці;
* номер будинку;
* місто;
* район.

Об’єкт лічильник має такі властивості:

* початкова дата;
* кінцева дата;
* показання на початку;
* показання наприкінці;
* номер лічильника.

3.2. Визначення взаємозв'язків між сутностями

Зв'язок “один до багатьох” у більшості випадків відображає реальний взаємозв'язок сутностей у предметній області. Вона реалізується парою “зовнішній ключ – первинний ключ”, тобто коли визначений зовнішній ключ, що посилається на первинний ключ іншої таблиці.

У поставленому завдання кожен клієнт постачальника газу може мати декілька лічильників. Тобто, між об’єктами КЛІЄНТ і ЛІЧИЛЬНИК є зв’язок «один до багатьох».

Взаємозв'язок між об'єктами є частиною концептуальної моделі і повинні відображатися в базі даних. Отримана після цього логічна модель представлена на рис. 1.

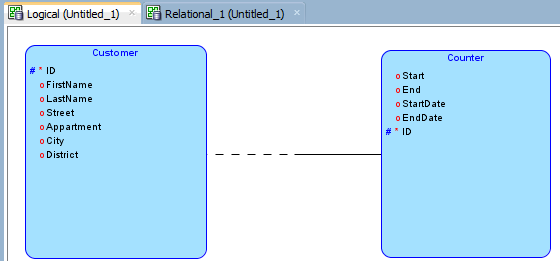


Рис. 1. Логічна модель бази даних «GAS»

3.3. Завдання первинних і альтернативних ключів, визначення атрибутів сутностей

Атрибут – це інформаційне відображення властивостей об'єкта. Кожен об'єкт характеризується своїми основними атрибутами.

За завданням курсової роботи необхідно створити таблиці – регулярні структури, що складаються з кінцевих наборів полів по вертикалі й однотипних записах по горизонталі.

Отже, об’єкт КЛІЄНТ можна ідентифікувати за унікальним числовим ключем, який буде присвоєний кожному клієнту свій. Усі інші характеристики – залишаться атрибутами цього об’єкту: ім’я, прізвище, назва вулиці домашньої адреси, номер дому, місто, район проживання.

У об’єкті ЛІЧИЛЬНИК унікальним ключем може слугувати його номер, що зазначений на усіх лічильникам. За ним можна буде перебирати усі ці об’єкти таблиці. Інші характеристики лічильника будуть слугувати атрибутами: початкова дата, кінцева дата, дані на початку, дані наприкінці.

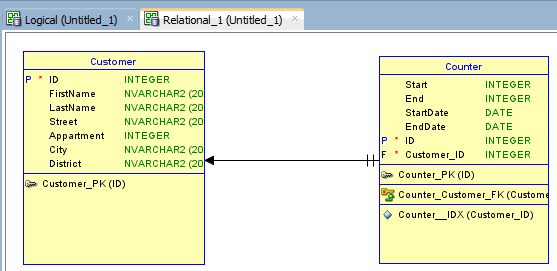


Рис. 2. ER-діаграма таблиці

Атрибути, які є складовими бази даних для розглянутої моделі, приведені в табл. 3.1.

Атрибути і первинні ключі сутностей інформаційної моделі **Таблиця 3.1**

|  |  |  |
| --- | --- | --- |
| **Сутність** | **Первинний ключ** | **Атрибути** |
| КЛІЄНТ | Унікальний цифровий ключ клієнта | Ім’я клієнта |
| Прізвище клієнта |
| Назва вулиці |
| Номер будинку |
| Місто |
| Район |
| ЛІЧИЛЬНИК | Унікальний номер лічильника | Початкова дата |
| Кінцева дата |
| Показання на початку |
| Показання наприкінці |

Генерація SQL-скрипт для створення таблиць, ключів, атрибутів та відношень наведено на рис. 3.

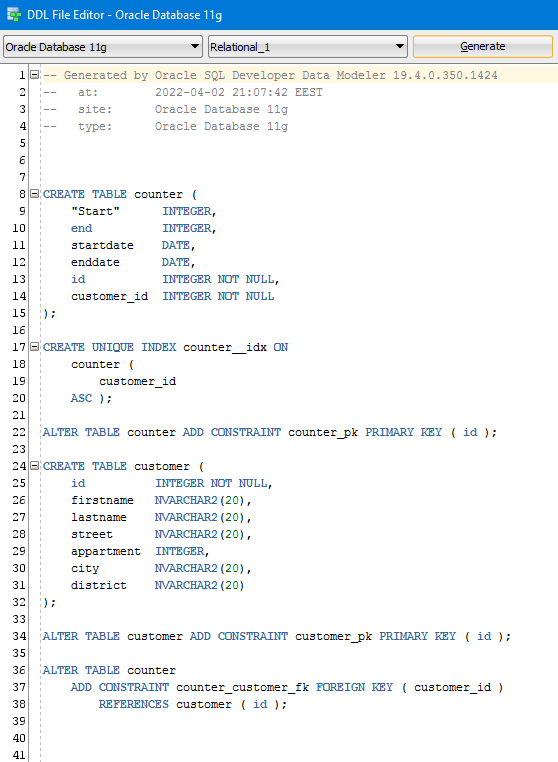


Рис. 3. Сгенерований SQL-скрипт

4. СТВОРЕННЯ ФІЗИЧНОЇ МОДЕЛІ

Як вже було зазначено вище, *Таблиця* – регулярна структура, що складається з кінцевого набору полів по вертикалі й однотипних записах по горизонталі.

Атрибут – це інформаційне відображення властивостей об'єкта. Кожен об'єкт характеризується своїми основними атрибутами.

При реалізації фізичної моделі бази даних, атрибут часто називають елементом даних, полем даних чи просто полем. Записи розташовані у рядках таблиць.

Запис даних – це сукупність значень зв’язаних елементів даних. Кожен запис таблиці складається однакового і кінцевого числа полів одного типу. Тип даних характеризує кожний елемент запису.

Поняття типу даних збігається з типи даних у мовах програмування. СУБД приймає значення у символьних, числових, бітових даних, даних дати та часу і т.п.

Призначення імен таблиць і їхніх атрибутів відбиті в у табл. 4.1.

Проект таблиць для фізичної моделі Таблиця 5.1.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Customer (КЛІЄНТ) | | | 1 | | |
| № п/п | Найменування стовбців | Примітка | Тип | | Розмір |
| 1. | ID | Унікальний цифровий ключ клієнта | INTEGER | |  |
| 2. | FIRSTNAME | Ім’я клієнта | NVARCHAR2 | | 20 |
| 3. | LASTNAME | Прізвище клієнта | NVARCHAR2 | | 20 |
| 4. | STREET | Назва вулиці | NVARCHAR2 | | 20 |
| 5. | APPARTMENT | Номер будинку | INTEGER | |  |
| 6. | CITY | Місто | NVARCHAR2 | | 20 |
| 7. | DISTRICT | Район | NVARCHAR2 | | 20 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Counter (ЛІЧИЛЬНИК) | | | 2 | | |
| № п/п | Найменування стовбців | Примітка | Тип | | Розмір |
| 1. | START\_N | Показання на початку | INTEGER | |  |
| 2. | END | Показання наприкінці | INTEGER | | 20 |
| 3. | STARTDATE | Початкова дата | DATE | | 20 |
| 4. | ENDDATE | Кінцева дата | DATE | | 20 |
| 5. | ID | Унікальний номер лічильника | INTEGER | |  |
| 6. | CUSTOMER\_ID | Ключ клієнта | INTEGER | | 20 |

5. ОПИС ПРОГРАМИ

5.1. Логічна структура

Логічна структура програми полягає в тому, що адміністратор підприємства, що займається постачанням газу має доступ до бази даних клієнтів та їх газових лічильників у вигляді таблиць. При цьому враховується, що кожен клієнт газового підприємства може мати декілька осель та декілька газових лічильників у них. Адміністратор бази даних має змогу додавати, редагувати, видаляти записи використовуючи програми.

При відкритті програми у адміністратора бази даних повинен бути вибір між редагуванням об’єктів: даних про клієнтів газового підприємства, або їх лічильників.

Після обрання об’єкту, у адміністратора є змога передивлятися записи, редагувати кожен їх атрибут, видаляти записи з таблиць, створювати нові, здійснювати пошук по атрибутам.

З кожного об’єкту є можливість створити звіт. Після цього, його можна або надрукувати, або зберегти у різні формати: .xls (Microsoft Excel), .pdf (Portable Document Format), .doc (Microsoft Word).

5.2. Екранні форми

Форма «Customers»

При розробці програми на першу форму був розміщений TableAdapter з таблицею клієнтів.

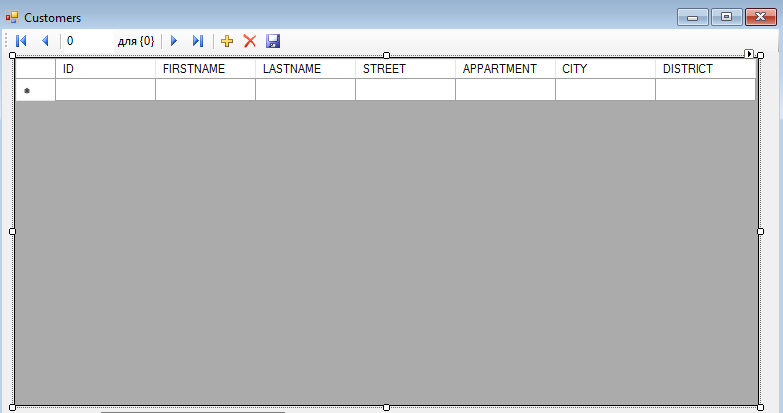


Рис. 4. CUSTOMERS TableAdapter на першій формі

Ця таблиця має такі стовбці:

* ID – унікальний цифровий ключ клієнта;
* FIRSTNAME – ім’я клієнта;
* LASTNAME – прізвище клієнта;
* STREET – назва вулиці;
* APPARTMENT – номер будинку;
* CITY – місто;
* DISTRICT – район.

Панель зверху форми надає змогу адміністратору бази даних переходити по записам у таблиці, створювати нові, видаляти та зберігати зміни.

Під таблицею були розміщені 6 полів TextBox з таблиці CUSTOMERS (усі, окрім ID). Додана кнопка «Add Customers» виконує вставку запису до таблиці.

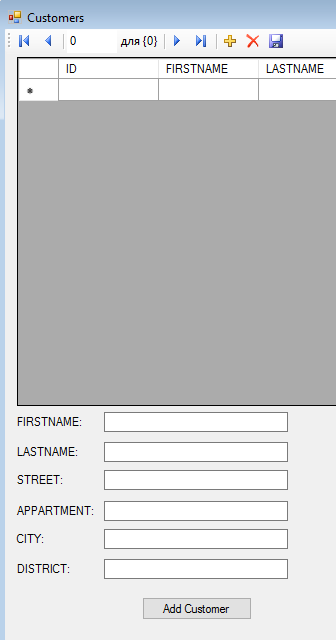


Рис. 5. Поля таблиці CUSTOMERS на першій формі

Для вставлення запису у таблицю необхідно створити метод, що буде приймати аргументи з TextBox та заносити запис у таблицю. Для цієї події був створений метод (запит) InsertCustomer(), що приймає наступні атрибути:

* :FIRSTNAME – ім’я клієнта;
* :LASTNAME – прізвище клієнта;
* :STREET – назва вулиці;
* :APPARTMENT – номер будинку;
* :CITY – місто;
* :DISTRICT – район.

Цей метод не приймає параметр ID клієнта, так як у базі даних була створена послідовність та тригер до неї, що генерує цифрові ключі для користувачів сам.

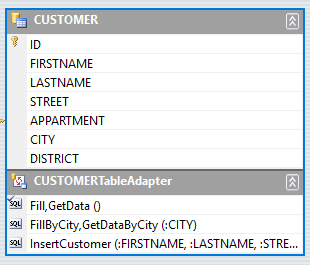


Рис. 6. Таблиця CUSTOMER у конструкторі наборів даних

На кнопці «Add Customer» був зроблений обробник події клацання по неї, який визиває метод InsertCustomer(), у який передається значення TextBox полів, що розташовані на формі. Для деяких полів, що мають числовий формат даних, застосований метод ToInt32() класу Convert, що переводить текстові значення чисел у формат даних Int32.

Після того, як спрацьовує метод InsertCustomer(), викликається метод за замовчуванням, який був створений Microsoft Visual Studio Fill(). Він заново заповнює TableAdapter CUSTOMER і дає змогу адміністратору бази даних газового постачальника побачити зміни у таблиці.

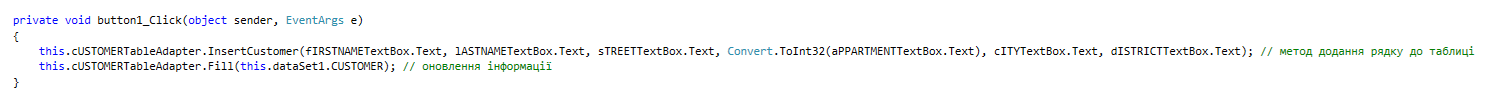


Рис. 7. Код обробки натискання кнопки «Add Customer»

З правої частини форми під CUSTOMER TableAdapter розташоване поле TextBox CITY для пошуку користувачів за їх містом. Знизу додана кнопка «Search by city», що виводить результат пошуку у TableAdapter.



Рис. 8. Права частина першої форми з інтерфейсом для пошуку клієнтів за містом

Для реалізації цієї пошукової системи, був створений метод FillByCity(), CUSTOMER TableAdapter, який приймає один єдиний аргумент: CITY (місто).

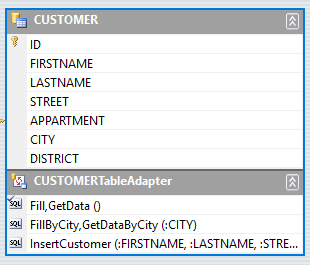


Рис. 9. Таблиця CUSTOMER у конструкторі наборів даних

У обробці події клацання по кнопці «Search by city» визивається метод FillByCity(), у параметрах якого передається значення другого поля TextBox для пошуку міста. Цей метод оновлює вигляд CUSTOMER TableAdapter та виводить відфільтровані результати.

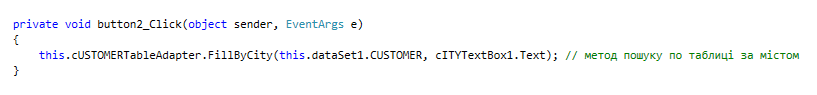


Рис. 10. Обробник натискання кнопки «Search by city»

По центру у нижній форми розташовані 2 кнопки: «Customers Report» та «Switch to Counters». Перша з них, «Customers Report» – відкриває іншу форму зі звітом по клієнтам компанії, що займається постачанням газу, а «Switch to Counters» – відкриває форму, що дає змогу перегляду таблиці газових лічильників користувачів. Для цього була створена Form2 та у обробці події кліку на кнопку «Switch to Counters» визваний метод Show(), що показує форму адміністратори бази даних.

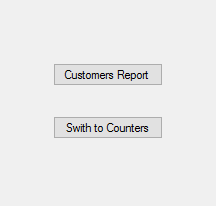


Рис. 11. Нижня центральна частина першої форми

Отже, фінальний вигляд першої форми для праці з таблицею CUSTOMER наведений на рис. 12.

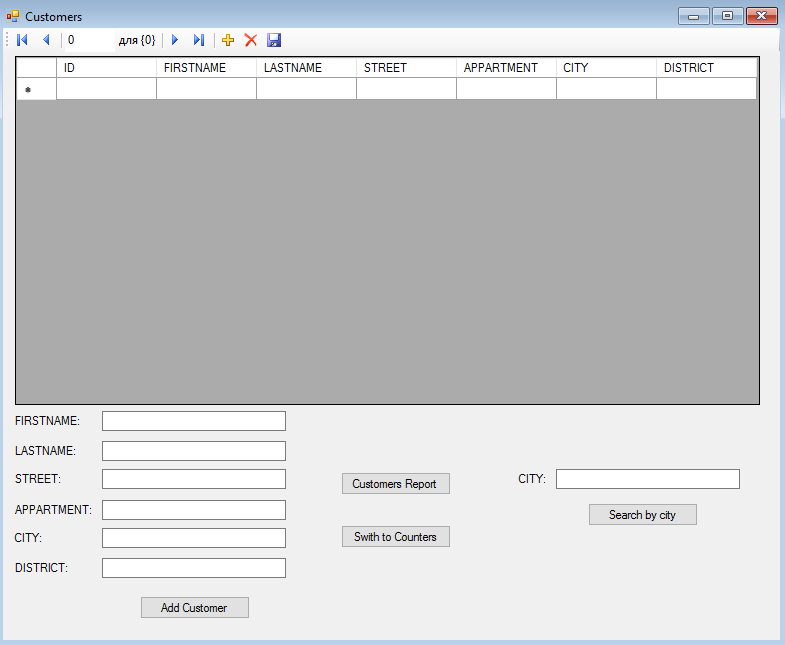


Рис. 12. Фінальний вигляд першої форми «Customers»

Форма «Counters»

Як було описано вище, після натискання кнопки «Switch to Counters» на першій формі, відкривається форма «Counters».

На формі розташований COUNTERS TableAdapter з таблицею про газові лічильники клієнтів.

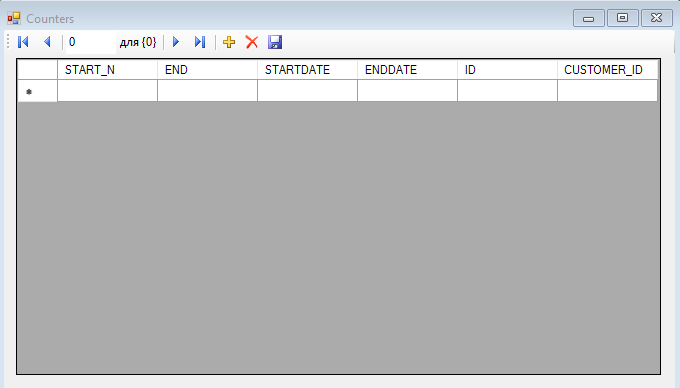


Рис. 13. COUNTERS TableAdapter на другій формі

Таблиця COUNTERS має наступні стовбці:

* START\_N – показання на початку;
* END – показання наприкінці;
* STARTDAT – початкова дата;
* ENDDATE – кінцева дата;
* ID – номер газового лічильника.
* CUSTOMER\_ID – ID користувача, якому належить газовий лічильник.

На цій формі також є панель, що дозволяє передивлятись записи у таблиці, створювати нові, видаляти записи та зберігати зміни у таблиці.

З лівої частини форми, під COUNTER TableAdapter розташовані усі полі атрибутів, які має об’єкт лічильника. Під ними створена кнопка, що додає запис у таблицю після введення даних у всі поля TextBox.

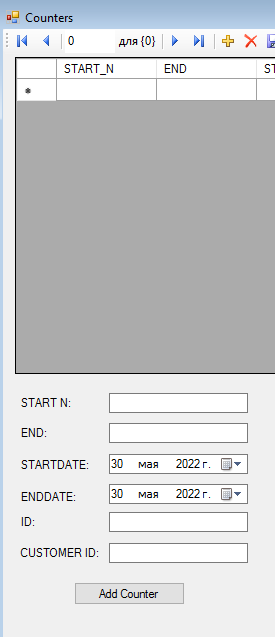


Рис. 14. Ліва частина форми «Counters»

Після заповнення полів лічильника, адміністратору бази слід натиснути на кнопку «Add Counter», яка визиває створений у конструкторі даних метод InsertCounter().

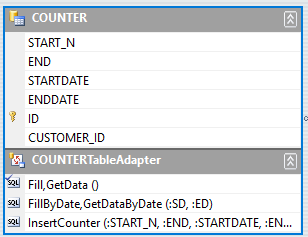


Рис. 15. Таблиця COUNTER у конструкторі наборів даних

Метод InsertCounter() приймає як аргументи усі поля TextBox газового лічильника. Для полів START\_N, END, ID, CUSTOMER\_ID застосований метод ToInt32() класу Convert, що переводить текстові значення чисел у формат даних Int32 для правильного занесення даних у таблицю.

Після того, як метод InsertCounter() був виконаний, викликається метод TableAdapter.Fill(). Він знову рендерить TableAdapter COUNTER відображає зміни у таблиці.

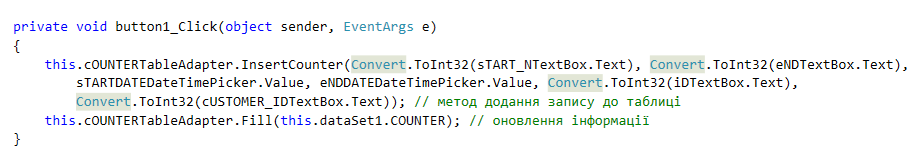


Рис. 16. Обробник натискання «Add Counter»

Аналогічно першій формі, у правій частині форми «Counters» розташований пошуковий інтерфейс. Пошук здійснюється по даті початку і закінчення терміну показників газових лічильників за допомогою елементів DateTimePicker. Під полями розташовано кнопка «Search by date».

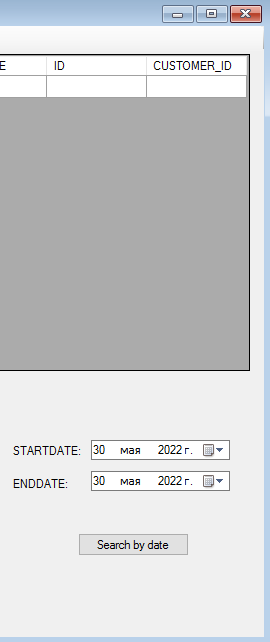


Рис. 17. Пошуковий інтерфейс на формі «Counters»

Був створений метод FillByDate() у конструкторі наборів даних. Він приймає 2 аргумента:

* STARTDATE – дата початку заміру показників на лічильнику;
* ENDDATE – дата кінця періоду замірів.

Обробник події «Search by date» визиває цей метод FillByDate(), у параметрах якого зазначені поля DateTimePicker. Після цього, викликається заздалегідь створений метод Fill(), що оновлює TableAdapter COUNTER.

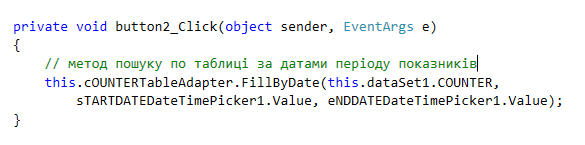


Рис. 18. Обробник натискання кнопки «Search by date»

Посередині, у нижній частині форми розташована кнопка «Counters Report», яка визиває метод Show() форми з інтерфейсом перегляду звітів.

Отже, фінальний вигляд форми «Counters» наведений на рис. 19:

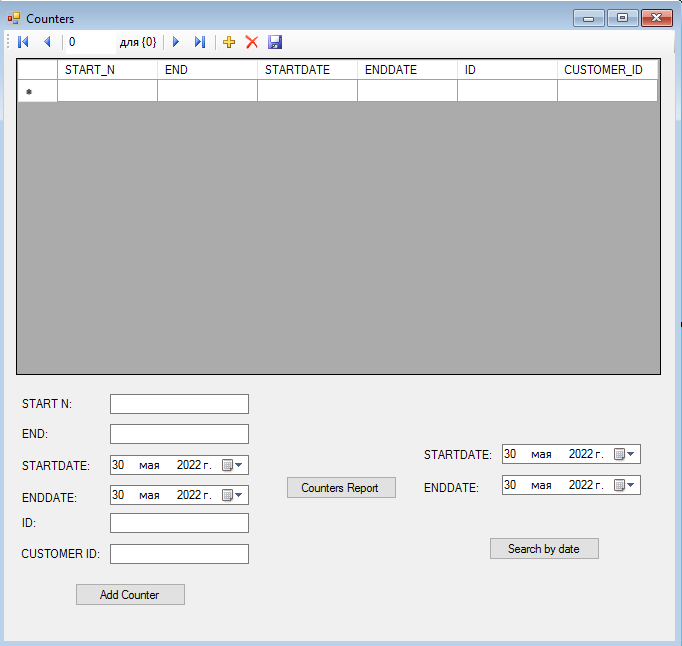


Рис. 19. Фінальний вигляд форми «Counters»

5.3. Звіти

Розроблена програма дозволяє генерувати, переглядати, друкувати та зберігати звіти з кожної таблиці розробленої бази даних для компанії-постачальника газу. Кожна форма зі звітом зверху має однаковий користувацький інтерфейс для виконання цих дій. Він був згенерований компонентом Microsoft Visual Studio – Report Viewer.



Рис. 20. Функціональний інтерфейс звітів

Звіт «Customers Report»

Звіт з клієнтів «Customers Report» має наступну структуру: він містить усі атрибути клієнтів: ім’я (FIRSTNAME), прізвище (LASTNAME), назва вулиці (STREET), номер будинку (APPARTMENT), місто (CITY), район (DISTRICT). Звіт надає змогу швидко зберегти інформацію про усіх клієнтів газового підприємства.



Рис. 21. Вигляд звіту «Customers Report»

Звіт «Counters Report»

Звіт газових лічильників «Counters Report» має наступну структуру: початкові показники лічильників усіх клієнтів (START\_N), кінцеві показники (END) їх сума («Expr»), враховуючи кожного клієнта, дата початку періоду (STARTDATE), кінця періоду (ENDDATE) кожного клієнта, індивідуальний номер кожного лічильника (ID), ідентифікатор клієнта, кому належить кожен лічильник (CUSTOMER\_ID). Звіт надає змогу адміністратору бази даних швидко порахувати витрати усіх клієнтів газової компанії.

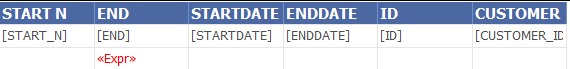


Рис. 22. Вигляд звіту «Customers Report»

5.4. Текст програми

**Form1.cs:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace GasProject

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

this.Validate();

this.cUSTOMERBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.dataSet1);

}

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

{

this.cUSTOMERTableAdapter.Fill(this.dataSet1.CUSTOMER); // заповнення таблиці при завантаженні форми

}

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

{

this.cUSTOMERTableAdapter.InsertCustomer(fIRSTNAMETextBox.Text, lASTNAMETextBox.Text, sTREETTextBox.Text, Convert.ToInt32(aPPARTMENTTextBox.Text), cITYTextBox.Text, dISTRICTTextBox.Text); // метод додання рядку до таблиці

this.cUSTOMERTableAdapter.Fill(this.dataSet1.CUSTOMER); // оновлення інформації

}

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

{

this.cUSTOMERTableAdapter.FillByCity(this.dataSet1.CUSTOMER, cITYTextBox1.Text); // метод пошуку по таблиці за містом

}

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

{

Form3 f3 = new Form3();

f3.Show(); // відображення форми зі звітом

}

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

{

Form2 f2 = new Form2();

f2.Show(); // відображення форми з таблицею про лічильники

}

}

}

**Form2.cs:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace GasProject

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

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

{

this.Validate();

this.cOUNTERBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.dataSet1);

}

private void Form2\_Load(object sender, EventArgs e)

{

this.cOUNTERTableAdapter.Fill(this.dataSet1.COUNTER); // заповнення таблиці при завантаженні форми

}

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

{

this.cOUNTERTableAdapter.InsertCounter(Convert.ToInt32(sTART\_NTextBox.Text), Convert.ToInt32(eNDTextBox.Text), sTARTDATEDateTimePicker.Value, eNDDATEDateTimePicker.Value, Convert.ToInt32(iDTextBox.Text), Convert.ToInt32(cUSTOMER\_IDTextBox.Text)); // метод додання запису до таблиці

this.cOUNTERTableAdapter.Fill(this.dataSet1.COUNTER); // оновлення інформації

}

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

{

this.cOUNTERTableAdapter.FillByDate(this.dataSet1.COUNTER, sTARTDATEDateTimePicker1.Value, eNDDATEDateTimePicker1.Value); // метод пошуку по таблиці за датами періоду показників

}

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

{

Form4 f4 = new Form4();

f4.Show(); // відображення форми з таблицею про лічильники

}

}

}

5.5. Список літератури

1. Add new data sources - Visual Studio (Windows): https://docs.microsoft.com/en-us/visualstudio/data-tools/add-new-data-sources?view=vs-2022 [Електронний ресурс]
2. Create and configure TableAdapters - Visual Studio (Windows): https://docs.microsoft.com/en-us/visualstudio/data-tools/create-and-configure-tableadapters?view=vs-2022 [Електронний ресурс]
3. Fill datasets by using TableAdapters - Visual Studio (Windows): https://docs.microsoft.com/en-us/visualstudio/data-tools/fill-datasets-by-using-tableadapters?view=vs-2022 [Електронний ресурс]
4. Insert new records into a database - Visual Studio (Windows): https://docs.microsoft.com/en-us/visualstudio/data-tools/insert-new-records-into-a-database?view=vs-2022 [Електронний ресурс]
5. Work with data in Visual Studio - Visual Studio (Windows): https://docs.microsoft.com/en-us/visualstudio/data-tools/accessing-data-in-visual-studio?view=vs-2022 [Електронний ресурс]
6. Рик Гринвальд, Роберт Стаковьяк, Гэри Додж, Дэвид Кляйн, Бен Шапиро, Кристофер Дж. Челья Программирование баз данных Oracle для профессионалов = Professional Oracle Programming — М.: «Диалектика», 2007. — 784 с. — ISBN 978-5-8459-1138-4.
7. Сервер Oracle 10G: Администрирование баз данных, Учебное пособие, Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.