НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»

ІНСТИТУТ КОМП'ЮТЕРНИХ СИСТЕМ

Кафедра інформаційних технологій

**КУРСОВИЙ ПРОЕКТ**

По курсу «Організація баз даних»

Варіант № 53

Виконав: студент 2 курсу, групи АД-231

Спеціальність 126 – «Інформаційні системи та технології»

Задорожний Віталій Віталійович

Керівник: доцент Гришин С.І.

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Кількість балів: \_\_\_\_\_\_\_ Оцінка: ECTS \_\_\_\_\_

Одеса – 2024

**ЗМІСТ**

[ЗАВДАННЯ ДО КУРСОВОЇ РОБОТИ 3](#_Toc184700148)

[ВСТУП 5](#_Toc184700149)

[ОСНОВНА ЧАСТИНА 6](#_Toc184700150)

[ВИСНОВОК 69](#_Toc184700151)

[ВИКОРИСТАНА ЛІТЕРАТУРА 70](#_Toc184700152)

# ЗАВДАННЯ ДО КУРСОВОЇ РОБОТИ

Завдання роботи - проектування та реалізація бази даних для заданої предметної області, організація доступу до даних з клієнтського додатку:

1. Обстеження предметної області с залученням інформаційних джерел. Включення до пояснювальної записки опису предметної області і переліку даних, що зберігаються з врахуванням завдання.

2. Визначення переліку бізнес-функцій системи. Проектування та включення до пояснювальної записки функціональної моделі (діаграми) системи і схеми бази даних. Створення і заповнення бази даних.( див. Лабораторні роботи №6, №11 )

3. Організація пошуку у таблицях бази даних з допомогою SQL запитів(див. Лабораторну роботу №8). Включення до пояснювальної записки.

4. Розрахунок підсумків у таблицях бази даних з допомогою агрегатних функцій(див. Лабораторну роботу №9). Включення до пояснювальної записки.

5. Пошук даних у складених сутностей з допомогою запитів на з’єднання та об’єднання таблиць (див. Лабораторну роботу №10). Включення до пояснювальної записки.

6. Модифікація даних з допомогою SQL запитів (див. Лабораторну роботу №11). Включення до пояснювальної записки.

7. Пошук даних у складених сутностей з допомогою підпорядкованих запитів (див. Лабораторну роботу №15). Включення до пояснювальної записки.

8. Використання уявлень і табличних виразів для обробки даних з допомогою (див. Лабораторну роботу №16). Включення до пояснювальної записки.

9. Використання параметрів при обробці даних з допомогою збережених процедур (див. Лабораторну роботу №18). Включення до пояснювальної записки.

10. Використання тригерів при обробці даних (див. Лабораторну роботу №19). Включення до пояснювальної записки.

11. Розробка клієнтської програми. Форми CRUD. (див. лабораторні роботи №№12-14, 20-24). Включення до пояснювальної записки.

12. Розробка клієнтської програми. Форми пошуку даних на клієнті. (див. Лабораторну роботу №23) Включення до пояснювальної записки.

13. Розробка клієнтської програми. Форми експорту даних у документи Word (див. Лабораторну роботу №24) Включення до пояснювальної записки.

# ВСТУП

Сучасний світ невпинно розвивається завдяки автоматизації та впровадженню інформаційних технологій у всі сфери людської діяльності. Однією з ключових складових ефективного функціонування організацій є здатність швидко обробляти, аналізувати та зберігати великі обсяги даних. Важливим інструментом для досягнення цієї мети є бази даних, які забезпечують централізоване управління даними, інтеграцію інформаційних систем та зручний доступ до необхідної інформації.

В умовах сучасних екологічних викликів та економічної необхідності зменшення енергоспоживання, особливо у житлових та комерційних будівлях, актуальність розробки систем моніторингу та аналізу споживання енергії є надзвичайно високою. Згідно з чинним законодавством, організації, відповідальні за управління енергоспоживанням, повинні забезпечувати ефективний контроль і оптимізацію використання енергетичних ресурсів.

Ця курсова робота присвячена проектуванню та реалізації частини програмного комплексу для обліку та аналізу енергоспоживання у багатоквартирних будинках. Основними завданнями є створення бази даних для зберігання інформації про енергоспоживання, проектування функціональної моделі системи, реалізація SQL-запитів для пошуку, аналізу та модифікації даних, а також розробка клієнтського додатку для взаємодії з базою даних.

Метою роботи є забезпечення ефективного моніторингу та аналізу енергоспоживання, що дозволить підвищити точність управління енергоресурсами, знизити витрати та сприяти досягненню енергетичної ефективності. Особливу увагу приділено можливості формування звітності, аналізу трендів споживання, а також розробці інструментів для експорту даних у форматах, зручних для подальшого використання.

# ОСНОВНА ЧАСТИНА

1. Обстеження предметної області с залученням інформаційних джерел. Включення до пояснювальної записки опису предметної області і переліку даних, що зберігаються з врахуванням завдання.

Система призначена комплексу для моніторингу та аналізу енергоспоживання в багатоквартирних будинках.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Поле | Тип | Розмір | Опис |
| 1 | HouseID | Текстовий | 20 | Унікальний ідентифікатор будівлі |
| 2 | Address | Текстовий | 60 | Адреса будівлі |
| 3 | District | Текстовий | 15 | Район міста |
| 4 | Land | Числовий | 10 | Площа прилеглої землі |
| 5 | Year | Числовий | 4 | Рік спорудження будівлі |
| 6 | Material | Текстовий | 15 | Матеріал стін будівлі |
| 7 | Floors | Числовий | 2 | Кількість поверхів у будівлі |
| 8 | Elevator | Логічний | 1 | Наявність ліфта |
| 9 | Flats | Числовий | 3 | Кількість квартир у будівлі |
| 10 | FlatID | Числовий | 4 | Номер квартири |
| 11 | Floor | Числовий | 2 | Номер поверху |
| 12 | Rooms | Числовий | 1 | Кількість кімнат у квартирі |
| 13 | Area | Числовий | Авто | Площа квартири |
| 14 | Residents | Числовий | Авто | Кількість мешканців у квартирі |
| 15 | Electricity | Числовий | Авто | Споживання електроенергії (Втг) |
| 16 | Water | Числовий | Авто | Споживання води (куб. м) |
| 17 | Heating | Числовий | Авто | Споживання опалення (Гкал) |
| 18 | Gas | Числовий | Авто | Споживання газу (куб. м) |
| 19 | HeatingSystem | Текстовий | 30 | Тип системи опалення |
| 20 | ElectricityCost | Грошовий | 15 | Витрати на електроенергію (грн) |
| 21 | WaterCost | Грошовий | 15 | Витрати на воду (грн) за місяць |
| 22 | HeatingCost | Грошовий | 15 | Витрати на опалення (грн) за місяць |
| 23 | GasCost | Грошовий | 15 | Витрати на газ (грн) за місяць |
| 24 | EnergyRating | Числовий | 2 | Енергетична ефективність будівлі |
| 25 | LastUpdate | Дата/час | Авто | Дата останнього оновлення даних |

2. Визначення переліку бізнес-функцій системи. Проектування та включення до пояснювальної записки функціональної моделі (діаграми) системи і схеми бази даних. Створення і заповнення бази даних.( див. Лабораторні роботи №№6,11 )

Діаграма розробленої бази даних на рисунку 1.

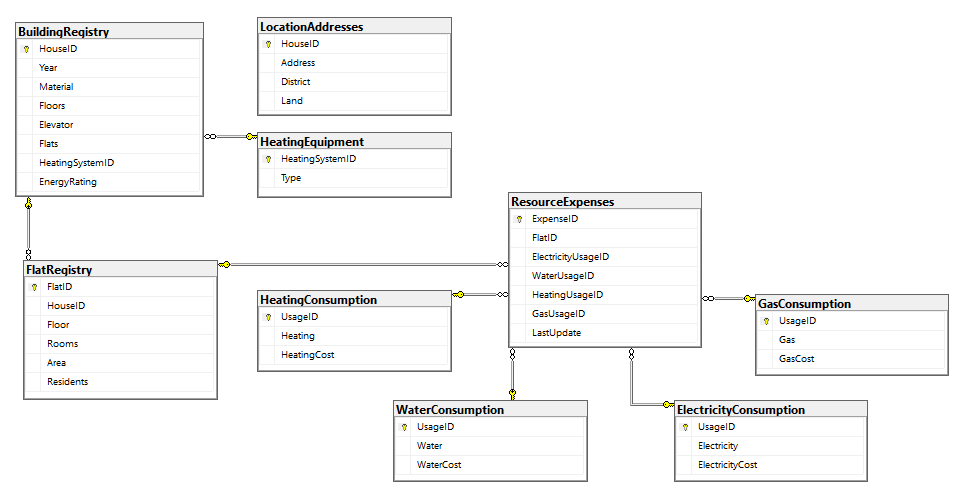


Рисунок 1 - Діаграма розробленої бази даних

Запит на заповнення таблиці [BuildingRegistry]

USE EnergyConsumption

INSERT INTO BuildingRegistry (HouseID, Year, Material, Floors, Elevator, Flats, HeatingSystemID, EnergyRating)

VALUES

('B011', 1991, 'Цегла', 6, 1, 22, 1, 78),

('B012', 1986, 'Панель', 10, 1, 38, 2, 63),

('B013', 2006, 'Бетон', 3, 1, 8, 3, 88),

('B014', 1979, 'Цегла', 7, 1, 26, 4, 52),

('B015', 2011, 'Піноблок', 9, 0, 34, 5, 92),

('B016', 2009, 'Панель', 11, 1, 42, 6, 83),

('B017', 1981, 'Цегла', 8, 1, 30, 1, 58),

('B018', 2016, 'Бетон', 10, 1, 38, 2, 98),

('B019', 1996, 'Цегла', 9, 0, 34, 3, 73),

('B020', 2001, 'Піноблок', 13, 0, 24, 4, 91);

Запит на заповнення таблиці [ElectricityConsumption]

Use EnergyConsumption

INSERT INTO [ElectricityConsumption] (Electricity, ElectricityCost) VALUES

(330, 850.00),

(490, 1268.00),

(270, 687.00),

(420, 1083.00),

(350, 903.00),

(310, 793.00),

(460, 1189.00),

(290, 740.00),

(390, 1004.00),

(230, 581.00),

(290, 740.00),

(380, 975.00),

(440, 1136.00),

(317, 811.00),

(300, 766.00),

(360, 925.00),

(430, 1109.00),

(305, 779.00),

(411, 1059.00),

(390, 1004.00),

(280, 713.00),

(330, 845.00),

(389, 1001.00),

(460, 1189.00),

(310, 793.00),

(370, 951.00),

(438, 1130.00),

(310, 793.00),

(386, 964.00),

(320, 819.00);

Запит на заповнення таблицы [FlatRegistry]

Use EnergyConsumption

INSERT INTO FlatRegistry(HouseID, Floor, Rooms, Area, Residents)

VALUES

('B001', 1, 5, 120, Null),

('B001', 3, 1, 40, 1),

('B001', 4, 2, 65, 2),

('B001', 5, 3, 80, 3),

('B001', 5, 2, 60, 2),

('B001', 2, 1, 45, 1),

('B001', 2, 2, 60, 1),

('B002', 6, 2, 70, 2),

('B002', 4, 4, 100, 3),

('B002', 1, 3, 80, Null),

('B002', 2, 2, 55, 1),

('B002', 3, 1, 40, 1),

('B002', 4, 4, 100, 3),

('B002', 5, 2, 70, 2),

('B003', 1, 4, 100, Null),

('B003', 2, 1, 38, 1),

('B004', 3, 3, 75, 2),

('B004', 5, 3, 80, 2),

('B005', 5, 2, 70, 2),

('B005', 7, 1, 45, 1),

('B006', 9, 3, 90, 3),

('B006', 10, 4, 120, 4),

('B007', 4, 2, 65, 2),

('B007', 6, 2, 68, 2),

('B008', 8, 4, 110, 4),

('B008', 9, 3, 95, 3),

('B009', 6, 2, 55, 2),

('B009', 7, 1, 50, 1),

('B010', 1, 6, 180, Null),

('B010', 11, 3, 85, 3)

Запит на заповнення таблиці [GasConsumption]

Use EnergyConsumption

INSERT INTO GasConsumption (Gas, GasCost)

VALUES

(120, 948.00),

(85, 671.50),

(150, 1185.00),

(110, 869.00),

(160, 1264.00),

(90, 711.00),

(180, 1422.00),

(130, 1033.00),

(140, 1106.00),

(100, 790.00),

(170, 1339.00),

(115, 904.50),

(190, 1501.00),

(95, 750.50),

(200, 1580.00),

(130, 1027.00),

(0, 0),

(0, 0),

(150, 1185.00),

(90, 711.00),

(160, 1264.00),

(120, 948.00),

(140, 1106.00),

(105, 827.50),

(195, 1539.50),

(125, 986.50),

(170, 1339.00),

(95, 750.50),

(185, 1458.50),

(115, 904.50)

Запит на заповнення таблиці [HeatingConsumption]

Use EnergyConsumption

INSERT INTO HeatingConsumption (Heating, HeatingCost) VALUES

(1.823, 3050.78),

(1.112, 1825.44),

(1.978, 3275.89),

(1.365, 2250.75),

(1.647, 2725.31),

(1.234, 2031.45),

(1.567, 2590.23),

(0.987, 1615.27),

(1.789, 2945.68),

(1.456, 2389.14),

(1.689, 2756.98),

(1.342, 2198.76),

(1.223, 1990.50),

(1.835, 3080.20),

(0, 0),

(0, 0),

(1.899, 3135.65),

(1.078, 1775.22),

(1.567, 2590.23),

(1.321, 2180.18),

(1.435, 2365.75),

(1.200, 1980.40),

(1.768, 2925.82),

(1.289, 2130.45),

(1.615, 2660.65),

(1.942, 3215.34),

(1.112, 1835.65),

(1.576, 2600.20),

(1.278, 2110.45),

(1.689, 2785.80)

Запит на заповнення таблицы [HeatingEquipment]

USE EnergyConsumption

insert into HeatingEquipment (HeatingSystemID, Type)

values

(1,'Центральне опалення'),

(2,'Інфрачервоне опалення'),

(3,'Теплові насоси'),

(4,'Печі та каміни'),

(5,'Підлоги з підігрівом'),

(6,'Сонячні системи опалення')

Запит на заповнення таблиці [LocationAdresses]

Use EnergyConsumption

INSERT INTO LocationAddresses (HouseID, Address, District, Land)

VALUES

('E011', 'вулиця Тараса Шевченка, 10', 'Центральний', 800),

('E012', 'вулиця Київська, 12', 'Приморський', 620),

('E013', 'вулиця Сонячна, 34', 'Малиновський', 770),

('E014', 'вулиця Академічна, 16', 'Приморський', 660),

('E015', 'вулиця Вишнева, 22', 'Центральний', 615),

('E016', 'проспект Миру, 45', 'Київський', 530),

('E017', 'вулиця Польова, 8', 'Приморський', 510),

('E018', 'бульвар Незалежності, 9', 'Приморський', 560),

('E019', 'вулиця Жовтнева, 14', 'Центральний', 610),

('E020', 'вулиця Квіткова, 3', 'Малиновський', 660);

Запит на заповнення таблиці [ResourceExpenses]

USE EnergyConsumption;

INSERT INTO ResourceExpenses(FlatID, ElectricityUsageID, WaterUsageID, HeatingUsageID, GasUsageID, LastUpdate)

VALUES

(101, 1, 1, 1, 1, '2024-04-11 08:00:00'),

(102, 2, 2, 2, 2, '2024-02-04 19:32:51'),

(103, 3, 3, 3, 3, '2024-12-03 11:45:44'),

(104, 4, 4, 4, 4, '2024-05-02 13:15:50'),

(105, 5, 5, 5, 5, '2024-01-03 16:36:00'),

(106, 6, 6, 6, 6, '2024-11-04 19:45:20'),

(107, 7, 7, 7, 7, '2024-01-20 20:40:01'),

(108, 8, 8, 8, 8, '2024-01-06 22:05:12'),

(109, 9, 9, 9, 9, '2024-04-02 23:34:00'),

(110, 10, 10, 10, 10, '2024-12-09 04:16:03'),

(111, 11, 11, 11, 11, '2024-02-21 08:45:41'),

(112, 12, 12, 12, 12, '2024-08-27 09:45:05'),

(113, 13, 13, 13, 13, '2024-01-30 11:14:02'),

(114, 14, 14, 14, 14, '2024-03-15 15:33:21'),

(115, 15, 15, 15, 15, '2024-11-09 15:45:24'),

(116, 16, 16, 16, 16, '2024-04-14 14:32:33'),

(117, 17, 17, 17, 17, '2024-09-16 20:15:59'),

(118, 18, 18, 18, 18, '2024-04-24 22:30:00'),

(119, 19, 19, 19, 19, '2024-09-21 23:43:51'),

(120, 20, 20, 20, 20, '2024-12-28 01:41:04'),

(121, 21, 21, 21, 21, '2024-11-15 08:31:02'),

(122, 22, 22, 22, 22, '2024-02-20 09:13:23'),

(123, 23, 23, 23, 23, '2024-05-17 10:05:14'),

(124, 24, 24, 24, 24, '2024-08-28 12:32:55'),

(125, 25, 25, 25, 25, '2024-09-25 15:45:48'),

(126, 26, 26, 26, 26, '2024-03-08 18:00:36'),

(127, 27, 27, 27, 27, '2024-02-12 20:15:32'),

(128, 28, 28, 28, 28, '2024-11-17 22:30:51'),

(129, 29, 29, 29, 29, '2024-12-01 23:45:50'),

(130, 30, 30, 30, 30, '2024-02-19 01:44:02'),

Запит на заповнення таблиці [ResourceExpenses]

Use EnergyConsumption

INSERT INTO WaterConsumption (Water, WaterCost) VALUES

(11, 555.55),

(16, 778.23),

(13, 638.34),

(20, 847.46),

(9, 431.18),

(25, 1168.99),

(14, 690.78),

(21, 1003.29),

(12, 581.49),

(10, 463.69),

(23, 1085.63),

(13, 605.93),

(18, 842.48),

(15, 716.52),

(9, 456.42),

(22, 1023.50),

(14, 663.66),

(19, 913.65),

(12, 596.65),

(11, 515.69),

(14, 681.89),

(17, 815.69),

(10, 504.54),

(24, 1113.89),

(16, 751.82),

(13, 601.81),

(20, 931.54),

(15, 710.93),

(11, 542.65),

(22, 1037.94)

3. Організація пошуку у таблицях бази даних з допомогою SQL запитів(див. Лабораторну роботу №8). Включення до пояснювальної записки.

Виведення адрес даних про будівель, що знаходяться в Малиновському районі:

USE EnergyConsumption

SELECT \*

From LocationAddresses

Where District like '%Малиновський%'

Виведення усіх даних про будівлі, збудовані після 2000 року:

USE EnergyConsumption

SELECT \* FROM Buildings

WHERE Year > 2000;

Отримати дані про будівлі, в яких є ліфт:

USE EnergyConsumption

SELECT Buildings.HouseID, Buildings.Year, Buildings.Material, Buildings.EnergyRating

FROM Buildings

WHERE Elevator=1;

4. Розрахунок підсумків у таблицях бази даних з допомогою агрегатних функцій(див. Лабораторну роботу №9). Включення до пояснювальної записки.

Показати кількість даних квартир на кожному поверсі будівлі:

USE EnergyConsumption

SELECT HouseID, Floor, COUNT(FlatID) AS NumberOfFlats

FROM FlatRegistry

GROUP BY HouseID, Floor

ORDER BY HouseID;

Вивести інформацію про кількість квартир за кількістю мешканців, які в ній проживають:

USE EnergyConsumption

SELECT Residents, COUNT(FlatID) AS NumberOfFlats

FROM FlatRegistry

Where Residents is not Null

GROUP BY Residents

ORDER BY Residents;

Підрахувати загальні витрати на енергоресурси для кожної квартири:

USE EnergyConsumption;

SELECT

FlatRegistry.FlatID,

FlatRegistry.Rooms,

FlatRegistry.Area,

FlatRegistry.Residents,

SUM(ElectricityConsumption.ElectricityCost + WaterConsumption.WaterCost + GasConsumption.GasCost + HeatingConsumption.HeatingCost) as TotalCost

FROM

FlatRegistry

INNER JOIN

ResourceExpenses ON FlatRegistry.FlatID = ResourceExpenses.FlatID

LEFT JOIN

ElectricityConsumption ON ResourceExpenses.ElectricityUsageID = ElectricityConsumption.UsageID

LEFT JOIN

WaterConsumption ON ResourceExpenses.WaterUsageID = WaterConsumption.UsageID

LEFT JOIN

GasConsumption ON ResourceExpenses.GasUsageID = GasConsumption.UsageID

LEFT JOIN

HeatingConsumption ON ResourceExpenses.HeatingUsageID = HeatingConsumption.UsageID

GROUP BY

FlatRegistry.FlatID,

FlatRegistry.Rooms,

FlatRegistry.Area,

FlatRegistry.Residents;

5. Пошук даних у складених сутностей з допомогою запитів на з’єднання та об’єднання таблиць (див. Лабораторну роботу №10). Включення до пояснювальної записки.

Додати до завдання 4.3 інформацію про тип системи опалення для кожної квартири:

USE EnergyConsumption;

SELECT

FlatRegistry.FlatID,

FlatRegistry.Rooms,

FlatRegistry.Area,

FlatRegistry.Residents,

ElectricityConsumption.Electricity,

ElectricityConsumption.ElectricityCost,

WaterConsumption.Water,

WaterConsumption.WaterCost,

GasConsumption.Gas,

GasConsumption.GasCost,

HeatingConsumption.Heating,

HeatingConsumption.HeatingCost,

SUM(ElectricityConsumption.ElectricityCost + WaterConsumption.WaterCost + GasConsumption.GasCost + HeatingConsumption.HeatingCost) AS TotalCost

FROM

FlatRegistry

INNER JOIN

ResourceExpenses ON FlatRegistry.FlatID = ResourceExpenses.FlatID

LEFT JOIN

ElectricityConsumption ON ResourceExpenses.ElectricityUsageID = ElectricityConsumption.UsageID

LEFT JOIN

WaterConsumption ON ResourceExpenses.WaterUsageID = WaterConsumption.UsageID

LEFT JOIN

GasConsumption ON ResourceExpenses.GasUsageID = GasConsumption.UsageID

LEFT JOIN

HeatingConsumption ON ResourceExpenses.HeatingUsageID = HeatingConsumption.UsageID

LEFT JOIN

LocationAddresses ON FlatRegistry.FlatID = FlatRegistry.FlatID

GROUP BY

FlatRegistry.FlatID,

FlatRegistry.Rooms,

FlatRegistry.Area,

FlatRegistry.Residents,

ElectricityConsumption.Electricity,

ElectricityConsumption.ElectricityCost,

WaterConsumption.Water,

WaterConsumption.WaterCost,

GasConsumption.Gas,

GasConsumption.GasCost,

HeatingConsumption.Heating,

HeatingConsumption.HeatingCost;

Отримати список всіх будинків разом з даними про адресу та про систему отоплення:

USE EnergyConsumption;

SELECT

BuildingRegistry.HouseID,

LocationAddresses.Address,

LocationAddresses.District,

HeatingEquipment.Type,

BuildingRegistry.EnergyRating,

BuildingRegistry.Year,

BuildingRegistry.Material

FROM

BuildingRegistry

INNER JOIN

LocationAddresses ON BuildingRegistry.HouseID = LocationAddresses.HouseID

LEFT JOIN

HeatingEquipment ON BuildingRegistry.HeatingSystemID = HeatingEquipment.HeatingSystemID;

Вивести інформацію про квартири, де використовують центральну систему опалення:

USE EnergyConsumption;

SELECT

FlatRegistry.FlatID,

FlatRegistry.Rooms,

FlatRegistry.Area,

FlatRegistry.Residents,

BuildingRegistry.HouseID,

HeatingEquipment.Type AS HeatingSystemType

FROM

FlatRegistry

LEFT JOIN

BuildingRegistry ON FlatRegistry.HouseID = BuildingRegistry.HouseID

LEFT JOIN

HeatingEquipment ON BuildingRegistry.HeatingSystemID = HeatingEquipment.HeatingSystemID

WHERE

HeatingEquipment.Type LIKE '%Центральне%'

GROUP BY

FlatRegistry.FlatID,

FlatRegistry.Rooms,

FlatRegistry.Area,

FlatRegistry.Residents,

BuildingRegistry.HouseID,

HeatingEquipment.Type;

6. Модифікація даних з допомогою SQL запитів (див. Лабораторну роботу №11). Включення до пояснювальної записки.

Запит на оновлення таблиці [ElectricityConsumption]

USE EnergyConsumption;

UPDATE EU

SET EU.Electricity = 311

FROM ElectricityConsumption EU

INNER JOIN ResourceExpenses RE ON EU.UsageID = RE.ElectricityUsageID

WHERE EU.UsageID = 30;

Запит на додавання в таблицю [LocationAddresses]

USE EnergyConsumption

insert into LocationAddresses (HouseID,Address,District,Land)

Values ('B011','вулиця Цегляна, 12','Київський',700)

Запит на видалення в таблиці [LocationAddresses]

Use EnergyConsumption

delete from LocationAddresses

Where HouseID='B011'

7. Пошук даних у складених сутностей з допомогою підпорядкованих запитів (див. Лабораторну роботу №15). Включення до пояснювальної записки.

Додати до завдання 4.3 інформацію про тип системи опалення для кожної квартири з допомогою підпорядкованих запитів:

USE EnergyConsumption;

SELECT FlatRegistry.FlatID, FlatRegistry.Rooms, FlatRegistry.Area, FlatRegistry.Residents,

(

SELECT SUM(ElectricityConsumption.ElectricityCost + WaterConsumption.WaterCost + GasConsumption.GasCost + HeatingConsumption.HeatingCost)

FROM ResourceExpenses

LEFT JOIN ElectricityConsumption ON ResourceExpenses.ElectricityUsageID = ElectricityConsumption.UsageID

LEFT JOIN WaterConsumption ON ResourceExpenses.WaterUsageID = WaterConsumption.UsageID

LEFT JOIN GasConsumption ON ResourceExpenses.GasUsageID = GasConsumption.UsageID

LEFT JOIN HeatingConsumption ON ResourceExpenses.HeatingUsageID = HeatingConsumption.UsageID

WHERE ResourceExpenses.FlatID = FlatRegistry.FlatID

) AS TotalCost ,

(

SELECT HeatingEquipment.Type

FROM BuildingRegistry

LEFT JOIN HeatingEquipment ON BuildingRegistry.HeatingSystemID = HeatingEquipment.HeatingSystemID

WHERE BuildingRegistry.HouseID = FlatRegistry.HouseID

) AS HeatingSystemType

FROM FlatRegistry;

Отримати список всіх будинків разом з даними про адресу з допомогою підпорядкованого запита, де площа прилеглої землі більше 600:

USE EnergyConsumption;

SELECT BuildingRegistry.HouseID, LocationAddresses.Address, LocationAddresses.District,

BuildingRegistry.Year, BuildingRegistry.Material

FROM BuildingRegistry

INNER JOIN LocationAddresses ON BuildingRegistry.HouseID = LocationAddresses.HouseID

WHERE LocationAddresses.HouseID IN (

SELECT HouseID

FROM LocationAddresses

WHERE Land > 600

);

Вивести інформацію про квартири, де використовують центральну систему опалення з допомогою підпорядкованого запита:

USE EnergyConsumption;

SELECT FlatID, Rooms, Area, Residents, HouseID, HeatingSystemType

FROM (

SELECT FlatRegistry.FlatID, FlatRegistry.Rooms, FlatRegistry.Area, FlatRegistry.Residents,

BuildingRegistry.HouseID, HeatingEquipment.Type AS HeatingSystemType

FROM FlatRegistry

LEFT JOIN BuildingRegistry ON FlatRegistry.HouseID = BuildingRegistry.HouseID

LEFT JOIN HeatingEquipment ON BuildingRegistry.HeatingSystemID = HeatingEquipment.HeatingSystemID

) AS SystemType

WHERE HeatingSystemType LIKE '%Центральне%';

8. Використання уявлень і табличних виразів для обробки даних з допомогою (див. Лабораторну роботу №16). Включення до пояснювальної записки.

Уявлення, що виводе інформацію про квартири, включаючи їхні характеристики:

USE EnergyConsumption;

GO

CREATE VIEW FlatsTotalCostView AS

SELECT

FlatRegistry.FlatID,

FlatRegistry.Rooms,

FlatRegistry.Area,

FlatRegistry.Residents,

ElectricityConsumption.Electricity,

ElectricityConsumption.ElectricityCost,

WaterConsumption.Water,

WaterConsumption.WaterCost,

GasConsumption.Gas,

GasConsumption.GasCost,

HeatingConsumption.Heating,

HeatingConsumption.HeatingCost,

SUM(ElectricityConsumption.ElectricityCost + WaterConsumption.WaterCost + GasConsumption.GasCost + HeatingConsumption.HeatingCost) AS TotalCost,

HeatingEquipment.Type AS HeatingSystemType

FROM FlatRegistry

INNER JOIN ResourceExpenses ON FlatRegistry.FlatID = ResourceExpenses.FlatID

LEFT JOIN ElectricityConsumption ON ResourceExpenses.ElectricityUsageID = ElectricityConsumption.UsageID

LEFT JOIN WaterConsumption ON ResourceExpenses.WaterUsageID = WaterConsumption.UsageID

LEFT JOIN GasConsumption ON ResourceExpenses.GasUsageID = GasConsumption.UsageID

LEFT JOIN HeatingConsumption ON ResourceExpenses.HeatingUsageID = HeatingConsumption.UsageID

LEFT JOIN BuildingRegistry ON FlatRegistry.HouseID = BuildingRegistry.HouseID

LEFT JOIN HeatingEquipment ON BuildingRegistry.HeatingSystemID = HeatingEquipment.HeatingSystemID

GROUP BY

FlatRegistry.FlatID,

FlatRegistry.Rooms,

FlatRegistry.Area,

FlatRegistry.Residents,

ElectricityConsumption.Electricity,

ElectricityConsumption.ElectricityCost,

WaterConsumption.Water,

WaterConsumption.WaterCost,

GasConsumption.Gas,

GasConsumption.GasCost,

HeatingConsumption.Heating,

HeatingConsumption.HeatingCost,

BuildingRegistry.HouseID,

HeatingEquipment.Type;

Уявлення, що показує квартири з типом системи опалення "Центральне":

USE EnergyConsumption;

GO

CREATE VIEW CentralHeatingFlatsView AS

SELECT

FlatRegistry.FlatID,

FlatRegistry.Rooms,

FlatRegistry.Area,

FlatRegistry.Residents,

BuildingRegistry.HouseID,

HeatingEquipment.Type AS HeatingSystemType

FROM FlatRegistry

LEFT JOIN BuildingRegistry ON FlatRegistry.HouseID = BuildingRegistry.HouseID

LEFT JOIN HeatingEquipment ON BuildingRegistry.HeatingSystemID = HeatingEquipment.HeatingSystemID

WHERE HeatingEquipment.Type LIKE '%Центральне%'

GROUP BY FlatRegistry.FlatID, FlatRegistry.Rooms, FlatRegistry.Area, FlatRegistry.Residents, BuildingRegistry.HouseID, HeatingEquipment.Type;

Отримання інформації про розрахункові витрати, пов'язані зі споживанням еноргоресурсів в квартирах:

USE EnergyConsumption;

WITH CalculatedCosts AS (

SELECT

FlatRegistry.FlatID,

FlatRegistry.Rooms,

FlatRegistry.Area,

FlatRegistry.Residents,

BuildingRegistry.HouseID,

HeatingEquipment.Type AS HeatingSystemType,

SUM(ElectricityConsumption.ElectricityCost + WaterConsumption.WaterCost + GasConsumption.GasCost + HeatingConsumption.HeatingCost) AS TotalCost

FROM FlatRegistry

INNER JOIN ResourceExpenses ON FlatRegistry.FlatID = ResourceExpenses.FlatID

LEFT JOIN ElectricityConsumption ON ResourceExpenses.ElectricityUsageID = ElectricityConsumption.UsageID

LEFT JOIN WaterConsumption ON ResourceExpenses.WaterUsageID = WaterConsumption.UsageID

LEFT JOIN GasConsumption ON ResourceExpenses.GasUsageID = GasConsumption.UsageID

LEFT JOIN HeatingConsumption ON ResourceExpenses.HeatingUsageID = HeatingConsumption.UsageID

LEFT JOIN BuildingRegistry ON FlatRegistry.HouseID = BuildingRegistry.HouseID

LEFT JOIN HeatingEquipment ON BuildingRegistry.HeatingSystemID = HeatingEquipment.HeatingSystemID

GROUP BY FlatRegistry.FlatID, FlatRegistry.Rooms, FlatRegistry.Area, FlatRegistry.Residents, BuildingRegistry.HouseID, HeatingEquipment.Type

)

SELECT \* FROM CalculatedCosts;

Отримання інформації про квартири з центральним опаленням. Дані збираються з таблиць реєстру квартир, будинків та обладнання для опалення, об'єднуючи їх на основі відповідних ідентифікаторів та фільтруючи за типом опалення:

USE EnergyConsumption;

WITH CentralHeatingFlats AS (

SELECT

FlatRegistry.FlatID,

FlatRegistry.Rooms,

FlatRegistry.Area,

FlatRegistry.Residents,

BuildingRegistry.HouseID,

HeatingEquipment.Type AS HeatingSystemType

FROM FlatRegistry

LEFT JOIN BuildingRegistry ON FlatRegistry.HouseID = BuildingRegistry.HouseID

LEFT JOIN HeatingEquipment ON BuildingRegistry.HeatingSystemID = HeatingEquipment.HeatingSystemID

WHERE HeatingEquipment.Type LIKE '%Центральне%'

GROUP BY FlatRegistry.FlatID, FlatRegistry.Rooms, FlatRegistry.Area, FlatRegistry.Residents, BuildingRegistry.HouseID, HeatingEquipment.Type

)

SELECT \* FROM CentralHeatingFlats;

9. Використання параметрів при обробці даних з допомогою збережених процедур (див. Лабораторну роботу №18). Включення до пояснювальної записки.

Процедура отримує параметр @Flat та повертає ідентифікатор квартири,

де ідентифікатор квартири аналогічний наданому параметру. Ця процедура була використувана в формі CRUD(завдання 11), для пошуку посимвольного потрібної квартири.

USE [EnergyConsumption]

GO

Create procedure [dbo].[takeFlatID]

@Flat varchar(10)

as

select FlatID

from Flats

where FlatID like '%'+@Flat+'%'

Реалізація роботи клієнтської програми на рисунку 2

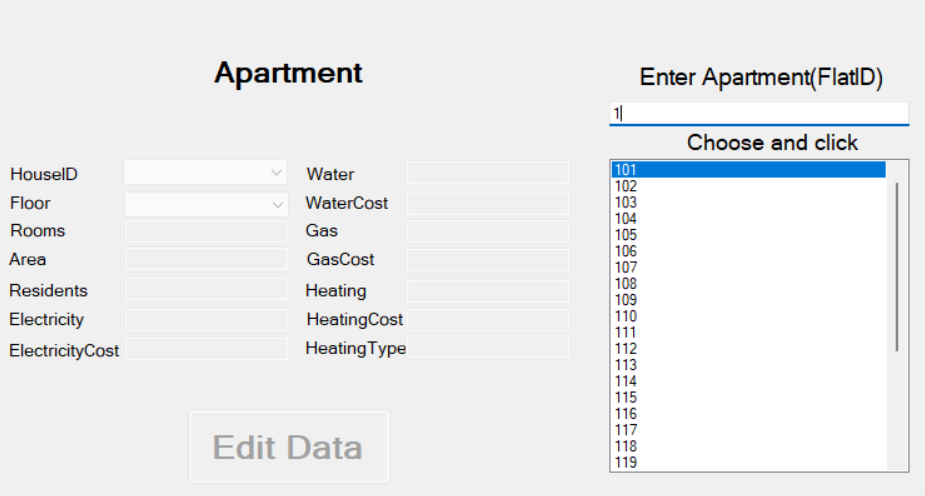


Рисунок 2 – Робота пошуку в програмі

Збережена процедура для пошуку даних в таблиці. Вхідний параметр процедури - повний ідентифікатор квартири. Вихідний набір даних містить id будинку, поверх, площу, кількість мешканців та використані енергоресурси квартири.

USE [EnergyConsumption]

GO

CREATE PROCEDURE searchFlat

@search\_Flat VARCHAR(10)

AS

BEGIN

SELECT

FlatRegistry.FlatID,

BuildingRegistry.HouseID,

FlatRegistry.Floor,

FlatRegistry.Rooms,

FlatRegistry.Area,

FlatRegistry.Residents,

ElectricityConsumption.Electricity,

ElectricityConsumption.ElectricityCost,

WaterConsumption.Water,

WaterConsumption.WaterCost,

GasConsumption.Gas,

GasConsumption.GasCost,

HeatingConsumption.Heating,

HeatingConsumption.HeatingCost,

HeatingEquipment.Type AS HeatingSystemType

FROM FlatRegistry

INNER JOIN ResourceExpenses ON FlatRegistry.FlatID = ResourceExpenses.FlatID

LEFT JOIN ElectricityConsumption ON ResourceExpenses.ElectricityUsageID = ElectricityConsumption.UsageID

LEFT JOIN WaterConsumption ON ResourceExpenses.WaterUsageID = WaterConsumption.UsageID

LEFT JOIN GasConsumption ON ResourceExpenses.GasUsageID = GasConsumption.UsageID

LEFT JOIN HeatingConsumption ON ResourceExpenses.HeatingUsageID = HeatingConsumption.UsageID

LEFT JOIN BuildingRegistry ON FlatRegistry.HouseID = BuildingRegistry.HouseID

LEFT JOIN HeatingEquipment ON BuildingRegistry.HeatingSystemID = HeatingEquipment.HeatingSystemID

WHERE FlatRegistry.FlatID = CONVERT(INT, @search\_Flat)

END

GO

Реалізація роботи клієнтської програми на рисунку 3

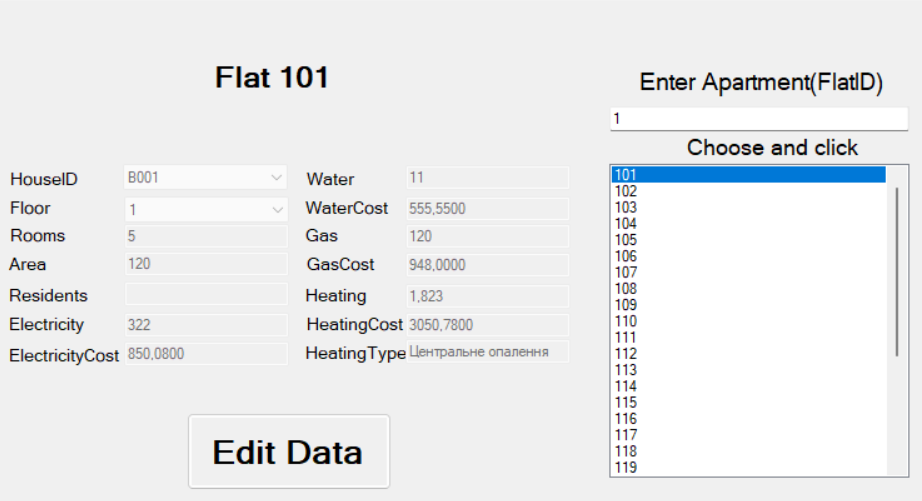


Рисунок 3 – Робота пошуку в програмі

10. Використання тригерів при обробці даних (див. Лабораторну роботу №19). Включення до пояснювальної записки.

Ці тригери оновляють дату в таблиці ResourseExpenses, якщо дані в таблицях ElectricityConsumption, WaterConsumption, HeatingConsumption, GasConsumption оновлюються.

USE EnergyConsumption

GO

CREATE TRIGGER UpdateDate\_Electricity

ON ElectricityConsumption

AFTER UPDATE

AS

BEGIN

SET NOCOUNT ON;

UPDATE ResourceExpenses

SET LastUpdate = GETDATE()

FROM inserted

WHERE ResourceExpenses.ElectricityUsageID = inserted.UsageID;

END;

GO

CREATE TRIGGER UpdateDate\_Gas

ON GasConsumption

AFTER UPDATE

AS

BEGIN

SET NOCOUNT ON;

UPDATE ResourceExpenses

SET LastUpdate = GETDATE()

FROM inserted

WHERE ResourceExpenses.GasUsageID = inserted.UsageID;

END;

GO

CREATE TRIGGER UpdateDate\_Water

ON WaterConsumption

AFTER UPDATE

AS

BEGIN

SET NOCOUNT ON;

UPDATE ResourceExpenses

SET LastUpdate = GETDATE()

FROM inserted

WHERE ResourceExpenses.WaterUsageID = inserted.UsageID;

END;

GO

CREATE TRIGGER UpdateDate\_Heating

ON HeatingConsumption

AFTER UPDATE

AS

BEGIN

SET NOCOUNT ON;

UPDATE ResourceExpenses

SET LastUpdate = GETDATE()

FROM inserted

WHERE ResourceExpenses.HeatingUsageID = inserted.UsageID;

END;

GO

11. Розробка клієнтської програми. Форми CRUD. (див.  лабораторні роботи №№12-14, 20-24). Включення до пояснювальної записки.

Головна форма клієнтської програми. В якої можна вибрати завдання:

Apartment Register – форма CRUD, Apartment Search - форма пошуку даних у клієнті, Export Data to Docx - форма експорту даних у документ Word.

Реалізація коду головної форми (MainForm.cs):

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace EnergyConsumption

{

public partial class MainForm : Form

{

private bool childFormOpen = false;

public MainForm()

{

InitializeComponent();

}

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

{

this.Close();

Application.Exit();

}

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

{

if (childFormOpen)

{

MessageBox.Show("Дочерняя форма уже открыта.");

return;

}

apartmentRegisterForm myApartmentRegisterForm = new apartmentRegisterForm();

myApartmentRegisterForm.FormClosed += ChildForm\_FormClosed;

myApartmentRegisterForm.Show();

this.Enabled = false;

childFormOpen = true;

}

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

{

if (childFormOpen)

{

MessageBox.Show("Дочерняя форма уже открыта.");

return;

}

apartmentSearchForm myApartmentSearchForm = new apartmentSearchForm();

myApartmentSearchForm.FormClosed += ChildForm\_FormClosed;

myApartmentSearchForm.Show();

this.Enabled = false;

childFormOpen = true;

}

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

{

if (childFormOpen)

{

MessageBox.Show("Дочерняя форма уже открыта.");

return;

}

exportForm myExportForm = new exportForm();

myExportForm.FormClosed += ChildForm\_FormClosed;

myExportForm.Show();

this.Enabled = false;

childFormOpen = true;

}

private void ChildForm\_FormClosed(object sender, FormClosedEventArgs e)

{

this.Enabled = true;

childFormOpen = false;

}

}

}

Реалізація головної форми на рисунку 4.

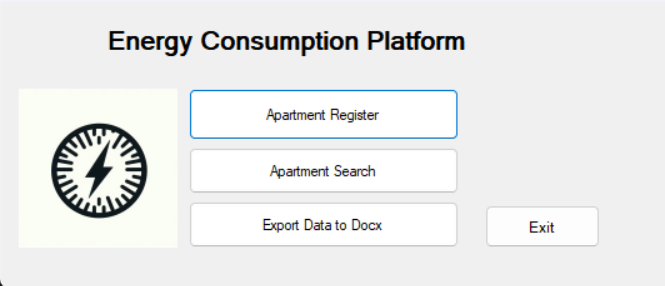


Рисунок 4 – Головна форма

Реалізація коду форми CRUD (apartmentRegisterForm.cs):

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 EnergyConsumption

{

public partial class apartmentRegisterForm : Form

{

private SqlConnection cnn;

String[] s = {"Rooms", "Area", "Residents", "Electricity", "ElectricityCost",

"Water", "WaterCost", "Gas", "GasCost", "Heating",

"HeatingCost","HeatingSystemType"};

Boolean editText = false;

public apartmentRegisterForm()

{

InitializeComponent();

}

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

{

cnn = new SqlConnection();

cnn.ConnectionString = "Data Source=(local); Initial Catalog = EnergyConsumption; Integrated Security=True";

foreach (Control control in this.Controls)

{

if (control is TextBox)

{

if (control == WaterCost || control == GasCost || control == ElectricityCost || control == Heating || control == HeatingCost)

{

((TextBox)control).KeyPress += TextBox\_KeyPress;

}

else

{

((TextBox)control).KeyPress += TextBox\_KeyPressValidation;

}

}

}

}

private void TextBox\_TextChanged(object sender, EventArgs e)

{

SqlCommand cmd = new SqlCommand("takeFlatID", cnn);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.Add("@Flat", SqlDbType.VarChar).Value = textBox1.Text;

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataTable dt = new DataTable();

da.Fill(dt);

listBox1.DataSource = dt;

listBox1.DisplayMember = dt.Columns[0].ColumnName;

}

private void listBox\_DoubleClick(object sender, EventArgs e)

{

SqlCommand cmd = new SqlCommand("searchFlat", cnn);

cmd.CommandType = CommandType.StoredProcedure;

DataRowView dr = (DataRowView)listBox1.SelectedItem;

cmd.Parameters.Add("@search\_Flat", SqlDbType.VarChar).Value =

dr.Row.ItemArray[0].ToString();

SqlDataAdapter da1 = new SqlDataAdapter(cmd);

DataTable dt = new DataTable();

da1.Fill(dt);

FlatText.Text = "Flat " + dr.Row.ItemArray[0].ToString();

fillComboBox();

comboBoxHouseID.Text = dt.Rows[0].ItemArray[1].ToString().Trim();

comboBoxFloor.Text = dt.Rows[0].ItemArray[2].ToString().Trim();

for (int i = 3; i < 15; i++)

{

TextBox textBox = this.Controls.Find(s[i - 3], true).FirstOrDefault() as TextBox;

textBox.Text = dt.Rows[0].ItemArray[i].ToString();

}

editButton.Enabled = true;

}

private void updateData()

{

SqlCommand cmd = new SqlCommand("takeFlatID", cnn);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.Add("@Flat", SqlDbType.VarChar).Value = textBox1.Text;

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataTable dt = new DataTable();

da.Fill(dt);

listBox1.DataSource = dt;

listBox1.DisplayMember = dt.Columns[0].ColumnName;

SqlCommand cmd1 = new SqlCommand("searchFlat", cnn);

cmd1.CommandType = CommandType.StoredProcedure;

DataRowView dr = (DataRowView)listBox1.SelectedItem;

cmd1.Parameters.Add("@search\_Flat", SqlDbType.VarChar).Value =

dr.Row.ItemArray[0].ToString();

SqlDataAdapter da1 = new SqlDataAdapter(cmd1);

DataTable dt1 = new DataTable();

da1.Fill(dt1);

FlatText.Text = "Flat " + dr.Row.ItemArray[0].ToString();

fillComboBox();

comboBoxHouseID.Text = dt1.Rows[0].ItemArray[1].ToString().Trim();

comboBoxFloor.Text = dt1.Rows[0].ItemArray[2].ToString().Trim();

for (int i = 3; i < 15; i++)

{

TextBox textBox = this.Controls.Find(s[i - 3], true).FirstOrDefault() as TextBox;

textBox.Text = dt.Rows[0].ItemArray[i].ToString();

}

}

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

{

buttonsEdit(true);

editButton.Hide();

changeDataButton.Show();

newDataButton.Show();

deleteDataButton.Show();

editText = true;

}

private void buttonsEdit(bool condition)

{

comboBoxHouseID.Enabled = condition;

comboBoxFloor.Enabled = condition;

for (int i = 3; i < 14; i++)

{

TextBox textBox = this.Controls.Find(s[i - 3], true).FirstOrDefault() as TextBox;

textBox.Enabled = condition;

}

}

private void fillComboBox()

{

SqlCommand cmdHouseID = new SqlCommand("SELECT DISTINCT HouseID FROM BuildingRegistry", cnn);

SqlDataAdapter daHouseID = new SqlDataAdapter(cmdHouseID);

DataTable dtHouseID = new DataTable();

daHouseID.Fill(dtHouseID);

foreach (DataRow row in dtHouseID.Rows)

{

row["HouseID"] = row["HouseID"].ToString().Trim();

}

comboBoxHouseID.DataSource = dtHouseID;

comboBoxHouseID.DisplayMember = "HouseID";

}

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

{

string selectedHouseID = comboBoxHouseID.SelectedItem?.ToString();

string type;

using (SqlConnection connection = new SqlConnection("Data Source=(local); Initial Catalog = EnergyConsumption; Integrated Security=True"))

{

connection.Open();

using (SqlCommand selectHeatingType = new SqlCommand("select Type from HeatingEquipment where HeatingEquipment.HeatingSystemID = (select HeatingSystemID from BuildingRegistry Where HouseID=@HouseID)", connection))

{

selectHeatingType.Parameters.AddWithValue("@HouseID", comboBoxHouseID.Text);

type = Convert.ToString(selectHeatingType.ExecuteScalar());

}

}

HeatingSystemType.Text = type;

if (type.Trim() == "Печі та каміни")

{

Heating.Text = "0";

HeatingCost.Text = "0";

}

else if (type.Trim() == "Сонячні системи опалення")

{

Gas.Text = "0";

GasCost.Text = "0";

}

}

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

{

newDataButton.Hide();

deleteDataButton.Hide();

changeDataButton.Hide();

confirmDataButton.Show();

textBox1.Enabled = false;

listBox1.Enabled = false;

FlatText.Text = "New Flat";

comboBoxHouseID.Text = "";

comboBoxFloor.Text = "";

for (int i = 3; i < 14; i++)

{

TextBox textBox = this.Controls.Find(s[i - 3], true).FirstOrDefault() as TextBox;

textBox.Text = "";

}

}

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

{

if (isEmpty())

{

MessageBox.Show("Fill in all the fields!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

int flatId;

using (SqlConnection connection = new SqlConnection("Data Source=(local); Initial Catalog = EnergyConsumption; Integrated Security=True"))

{

connection.Open();

using (SqlCommand insertIntoFlats = new SqlCommand("INSERT INTO FlatRegistry ( HouseID, Floor, Rooms, Area, Residents) OUTPUT INSERTED.FlatID VALUES (@HouseID, @Floor, @Rooms, @Area, @Residents);", connection))

{

insertIntoFlats.Parameters.AddWithValue("@HouseID", comboBoxHouseID.Text.Trim());

insertIntoFlats.Parameters.AddWithValue("@Floor", Convert.ToInt32(comboBoxFloor.Text));

insertIntoFlats.Parameters.AddWithValue("@Rooms", Convert.ToInt32(Rooms.Text));

insertIntoFlats.Parameters.AddWithValue("@Area", Convert.ToInt32(Area.Text));

insertIntoFlats.Parameters.AddWithValue("@Residents", Convert.ToInt32(Residents.Text));

flatId = Convert.ToInt32(insertIntoFlats.ExecuteScalar());

}

int electricityUsageId;

using (SqlCommand insertIntoElectricityUsage = new SqlCommand("INSERT INTO ElectricityConsumption (Electricity, ElectricityCost) OUTPUT INSERTED.UsageID VALUES (@Electricity, @ElectricityCost);", connection))

{

insertIntoElectricityUsage.Parameters.AddWithValue("@Electricity", Convert.ToInt32(Electricity.Text));

insertIntoElectricityUsage.Parameters.AddWithValue("@ElectricityCost", Convert.ToDouble(ElectricityCost.Text));

electricityUsageId = Convert.ToInt32(insertIntoElectricityUsage.ExecuteScalar());

}

int waterUsageid;

using (SqlCommand insertIntoWaterUsage = new SqlCommand("INSERT INTO WaterConsumption (Water, WaterCost) OUTPUT INSERTED.UsageID VALUES (@Water, @WaterCost);", connection))

{

insertIntoWaterUsage.Parameters.AddWithValue("@Water", Convert.ToInt32(Water.Text));

insertIntoWaterUsage.Parameters.AddWithValue("@WaterCost", Convert.ToDouble(WaterCost.Text));

waterUsageid = Convert.ToInt32(insertIntoWaterUsage.ExecuteScalar());

}

int heatingUsageid;

using (SqlCommand insertIntoHeatingUsage = new SqlCommand("INSERT INTO HeatingConsumption (Heating, HeatingCost) OUTPUT INSERTED.UsageID VALUES (@Heating, @HeatingCost);", connection))

{

insertIntoHeatingUsage.Parameters.AddWithValue("@Heating", Convert.ToDouble(Heating.Text));

insertIntoHeatingUsage.Parameters.AddWithValue("@HeatingCost", Convert.ToDouble(HeatingCost.Text));

heatingUsageid = Convert.ToInt32(insertIntoHeatingUsage.ExecuteScalar());

}

int gasUsageid;

using (SqlCommand insertIntoGasUsage = new SqlCommand("INSERT INTO GasConsumption (Gas, GasCost) OUTPUT INSERTED.UsageID VALUES (@Gas, @GasCost);", connection))

{

insertIntoGasUsage.Parameters.AddWithValue("@Gas", Convert.ToInt32(Gas.Text));

insertIntoGasUsage.Parameters.AddWithValue("@GasCost", Convert.ToDouble(GasCost.Text));

gasUsageid = Convert.ToInt32(insertIntoGasUsage.ExecuteScalar());

}

using (SqlCommand insertIntoResourceUsage = new SqlCommand("INSERT INTO ResourceExpenses (FlatID, ElectricityUsageID, WaterUsageID, HeatingUsageID, GasUsageID, LastUpdate) VALUES (@FlatID, @ElectricityUsageID, @WaterUsageID, @HeatingUsageID, @GasUsageID, GETDATE());", connection))

{

insertIntoResourceUsage.Parameters.AddWithValue("@FlatID", flatId);

insertIntoResourceUsage.Parameters.AddWithValue("@ElectricityUsageID", electricityUsageId);

insertIntoResourceUsage.Parameters.AddWithValue("@WaterUsageID", waterUsageid);

insertIntoResourceUsage.Parameters.AddWithValue("@HeatingUsageID", heatingUsageid);

insertIntoResourceUsage.Parameters.AddWithValue("@GasUsageID", gasUsageid);

insertIntoResourceUsage.ExecuteNonQuery();

}

}

MessageBox.Show("A new flat has been created!", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

FlatText.Text = "Flat " + flatId;

textBox1.Text = Convert.ToString(flatId);

textBox1.Enabled = true;

listBox1.Enabled = true;

newDataButton.Show();

deleteDataButton.Show();

changeDataButton.Show();

confirmDataButton.Hide();

}

}

private bool isEmpty()

{

return (string.IsNullOrWhiteSpace(comboBoxHouseID.Text) ||

string.IsNullOrWhiteSpace(comboBoxFloor.Text) ||

string.IsNullOrWhiteSpace(Rooms.Text) ||

string.IsNullOrWhiteSpace(Area.Text) ||

string.IsNullOrWhiteSpace(Electricity.Text) ||

string.IsNullOrWhiteSpace(ElectricityCost.Text) ||

string.IsNullOrWhiteSpace(Water.Text) ||

string.IsNullOrWhiteSpace(WaterCost.Text) ||

string.IsNullOrWhiteSpace(Heating.Text) ||

string.IsNullOrWhiteSpace(HeatingCost.Text) ||

string.IsNullOrWhiteSpace(Gas.Text) ||

string.IsNullOrWhiteSpace(GasCost.Text));

}

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

{

if (isEmpty())

{

MessageBox.Show("Fill in all the fields!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

DataRowView dr = (DataRowView)listBox1.SelectedItem;

int flatID = Convert.ToInt32(dr.Row.ItemArray[0]);

// Проверяем и конвертируем значения из полей

if (!int.TryParse(comboBoxFloor.Text, out int floor))

{

MessageBox.Show("Please enter a valid number for the floor.", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!int.TryParse(Rooms.Text, out int rooms))

{

MessageBox.Show("Please enter a valid number for rooms.", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!int.TryParse(Area.Text, out int area))

{

MessageBox.Show("Please enter a valid number for area.", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!int.TryParse(Residents.Text, out int residents))

{

MessageBox.Show("Please enter a valid number for residents.", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!int.TryParse(Water.Text, out int water))

{

MessageBox.Show("Please enter a valid number for water usage.", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!double.TryParse(WaterCost.Text, out double waterCost))

{

MessageBox.Show("Please enter a valid number for water cost.", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!int.TryParse(Gas.Text, out int gas))

{

MessageBox.Show("Please enter a valid number for gas usage.", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!double.TryParse(GasCost.Text, out double gasCost))

{

MessageBox.Show("Please enter a valid number for gas cost.", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!double.TryParse(Heating.Text, out double heating))

{

MessageBox.Show("Please enter a valid number for heating usage.", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!double.TryParse(HeatingCost.Text, out double heatingCost))

{

MessageBox.Show("Please enter a valid number for heating cost.", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!int.TryParse(Electricity.Text, out int electricity))

{

MessageBox.Show("Please enter a valid number for electricity usage.", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!double.TryParse(ElectricityCost.Text, out double electricityCost))

{

MessageBox.Show("Please enter a valid number for electricity cost.", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

using (SqlConnection connection = new SqlConnection("Data Source=(local); Initial Catalog = EnergyConsumption; Integrated Security=True"))

{

connection.Open();

using (SqlCommand updateFlatsCmd = new SqlCommand("UPDATE FlatRegistry SET HouseID = @HouseID, Floor = @Floor, Rooms = @Rooms, Area = @Area, Residents = @Residents WHERE FlatID = @FlatID", connection))

{

updateFlatsCmd.Parameters.AddWithValue("@FlatID", flatID);

updateFlatsCmd.Parameters.AddWithValue("@HouseID", comboBoxHouseID.Text.Trim());

updateFlatsCmd.Parameters.AddWithValue("@Floor", floor);

updateFlatsCmd.Parameters.AddWithValue("@Rooms", rooms);

updateFlatsCmd.Parameters.AddWithValue("@Area", area);

updateFlatsCmd.Parameters.AddWithValue("@Residents", residents);

updateFlatsCmd.ExecuteNonQuery();

}

using (SqlCommand updateWaterUsageCmd = new SqlCommand("UPDATE WaterConsumption SET Water = @Water, WaterCost = @WaterCost WHERE UsageID = (SELECT WaterUsageID FROM ResourceExpenses WHERE FlatID = @FlatID)", connection))

{

updateWaterUsageCmd.Parameters.AddWithValue("@FlatID", flatID);

updateWaterUsageCmd.Parameters.AddWithValue("@Water", water);

updateWaterUsageCmd.Parameters.AddWithValue("@WaterCost", waterCost);

updateWaterUsageCmd.ExecuteNonQuery();

}

using (SqlCommand updateGasUsageCmd = new SqlCommand("UPDATE GasConsumption SET Gas = @Gas, GasCost = @GasCost WHERE UsageID = (SELECT GasUsageID FROM ResourceExpenses WHERE FlatID = @FlatID)", connection))

{

updateGasUsageCmd.Parameters.AddWithValue("@FlatID", flatID);

updateGasUsageCmd.Parameters.AddWithValue("@Gas", gas);

updateGasUsageCmd.Parameters.AddWithValue("@GasCost", gasCost);

updateGasUsageCmd.ExecuteNonQuery();

}

using (SqlCommand updateHeatingUsageCmd = new SqlCommand("UPDATE HeatingConsumption SET Heating = @Heating, HeatingCost = @HeatingCost WHERE UsageID = (SELECT HeatingUsageID FROM ResourceExpenses WHERE FlatID = @FlatID)", connection))

{

updateHeatingUsageCmd.Parameters.AddWithValue("@FlatID", flatID);

updateHeatingUsageCmd.Parameters.AddWithValue("@Heating", heating);

updateHeatingUsageCmd.Parameters.AddWithValue("@HeatingCost", heatingCost);

updateHeatingUsageCmd.ExecuteNonQuery();

}

using (SqlCommand updateElectricityUsageCmd = new SqlCommand("UPDATE ElectricityConsumption SET Electricity = @Electricity, ElectricityCost = @ElectricityCost WHERE UsageID = (SELECT ElectricityUsageID FROM ResourceExpenses WHERE FlatID = @FlatID)", connection))

{

updateElectricityUsageCmd.Parameters.AddWithValue("@FlatID", flatID);

updateElectricityUsageCmd.Parameters.AddWithValue("@Electricity", electricity);

updateElectricityUsageCmd.Parameters.AddWithValue("@ElectricityCost", electricityCost);

updateElectricityUsageCmd.ExecuteNonQuery();

}

}

MessageBox.Show("Data changed!", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

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

{

if (isEmpty())

{

MessageBox.Show("Fill in all the fields!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

if (MessageBox.Show("Do you want to delete " + FlatText.Text + "?", "Are you sure?", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

DataRowView dr = (DataRowView)listBox1.SelectedItem;

int flatID = Convert.ToInt32(dr.Row.ItemArray[0]);

using (SqlConnection connection = new SqlConnection("Data Source=(local); Initial Catalog = EnergyConsumption; Integrated Security=True"))

{

connection.Open();

int expensesid;

using (SqlCommand SelectExpensesCmd = new SqlCommand("SELECT ExpenseID FROM ResourceExpenses WHERE FlatID = @FlatID", connection))

{

SelectExpensesCmd.Parameters.AddWithValue("@FlatID", flatID);

expensesid = Convert.ToInt32(SelectExpensesCmd.ExecuteScalar());

}

using (SqlCommand deleteExpensesCmd = new SqlCommand("DELETE FROM ResourceExpenses WHERE FlatID = @FlatID", connection))

{

deleteExpensesCmd.Parameters.AddWithValue("@FlatID", flatID);

deleteExpensesCmd.ExecuteNonQuery();

}

using (SqlCommand deleteFlatsCmd = new SqlCommand("DELETE FROM FlatRegistry WHERE FlatID = @FlatID", connection))

{

deleteFlatsCmd.Parameters.AddWithValue("@FlatID", flatID);

deleteFlatsCmd.ExecuteNonQuery();

}

using (SqlCommand deleteElectricityUsageCmd = new SqlCommand("DELETE FROM ElectricityConsumption WHERE UsageID = @ExpenseID", connection))

{

deleteElectricityUsageCmd.Parameters.AddWithValue("@ExpenseID", expensesid);

deleteElectricityUsageCmd.ExecuteNonQuery();

}

using (SqlCommand deleteWaterUsageCmd = new SqlCommand("DELETE FROM WaterConsumption WHERE UsageID = @ExpenseID", connection))

{

deleteWaterUsageCmd.Parameters.AddWithValue("@ExpenseID", expensesid);

deleteWaterUsageCmd.ExecuteNonQuery();

}

using (SqlCommand deleteGasUsageCmd = new SqlCommand("DELETE FROM GasConsumption WHERE UsageID = @ExpenseID", connection))

{

deleteGasUsageCmd.Parameters.AddWithValue("@ExpenseID", expensesid);

deleteGasUsageCmd.ExecuteNonQuery();

}

using (SqlCommand deleteHeatingUsageCmd = new SqlCommand("DELETE FROM HeatingConsumption WHERE UsageID = @ExpenseID", connection))

{

deleteHeatingUsageCmd.Parameters.AddWithValue("@ExpenseID", expensesid);

deleteHeatingUsageCmd.ExecuteNonQuery();

}

using (SqlCommand deleteHeatingEquipmentCmd = new SqlCommand("DELETE FROM HeatingEquipment WHERE HeatingSystemID = (SELECT HeatingSystemID FROM BuildingRegistry WHERE HouseID = (SELECT HouseID FROM FlatRegistry WHERE FlatID = @FlatID))", connection))

{

deleteHeatingEquipmentCmd.Parameters.AddWithValue("@FlatID", flatID);

deleteHeatingEquipmentCmd.ExecuteNonQuery();

}

MessageBox.Show("Flat deleted!", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

textBox1.Text = "1";

updateData();

}

}

}

}

private void comboBoxHouseID\_TextChanged(object sender, EventArgs e)

{

int numberOfFloor = 0;

using (SqlConnection connection = new SqlConnection("Data Source=(local); Initial Catalog = EnergyConsumption; Integrated Security=True"))

{

connection.Open();

using (SqlCommand cmdHouseID = new SqlCommand("SELECT Floors FROM BuildingRegistry WHERE HouseID = @HouseID", connection))

{

cmdHouseID.Parameters.AddWithValue("@HouseID", comboBoxHouseID.Text);

numberOfFloor = Convert.ToInt32(cmdHouseID.ExecuteScalar());

}

}

comboBoxFloor.Items.Clear();

for (int i = 1; i <= numberOfFloor; i++)

{

comboBoxFloor.Items.Add(i);

}

}

private void Electricity\_TextChanged(object sender, EventArgs e)

{

if (editText)

{

double Rate = 2.64;

if (double.TryParse(Electricity.Text, out double electricityValue))

{

double result = electricityValue \* Rate;

ElectricityCost.Text = result.ToString();

}

}

}

private void Water\_TextChanged(object sender, EventArgs e)

{

if (editText)

{

double Rate = 41.136;

if (double.TryParse(Water.Text, out double waterValue))

{

double result = waterValue \* Rate;

WaterCost.Text = result.ToString();

}

}

}

private void Gas\_TextChanged(object sender, EventArgs e)

{

if (editText)

{

double Rate = 7.99;

if (double.TryParse(Gas.Text, out double gasValue))

{

double result = gasValue \* Rate;

GasCost.Text = result.ToString();

}

}

}

private void Heating\_TextChanged(object sender, EventArgs e)

{

if (editText)

{

double Rate = 1813.94;

if (double.TryParse(Heating.Text, out double heatingValue))

{

double result = heatingValue \* Rate;

HeatingCost.Text = result.ToString();

}

}

}

private void TextBox\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && (e.KeyChar != ','))

{

e.Handled = true;

}

if ((e.KeyChar == ',') && (((TextBox)sender).Text.IndexOf(',') > -1))

{

e.Handled = true;

}

}

private void TextBox\_KeyPressValidation(object sender, KeyPressEventArgs e)

{

if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))

{

e.Handled = true;

}

}

private void label1\_Click(object sender, EventArgs e)

{

}

}

}

Реалізація форми СRUD на рисунку 5.

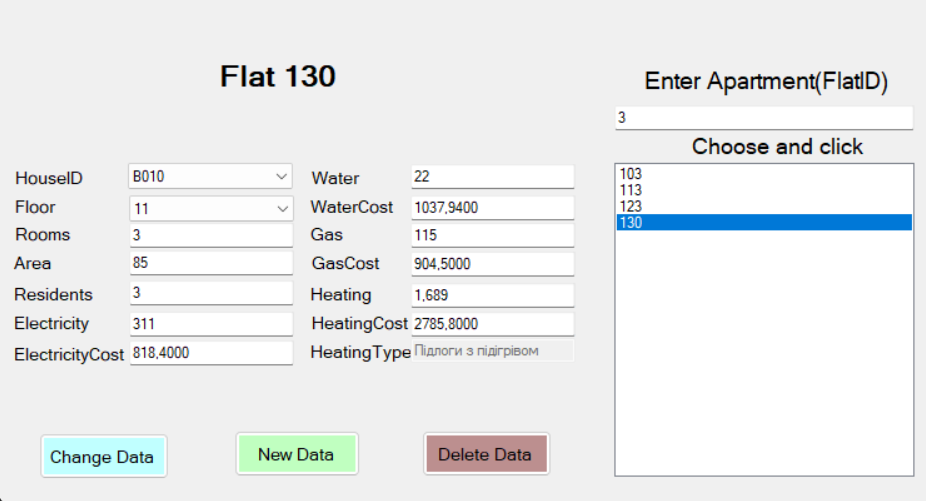


Рисунок 5 – Реалізація функціоналу форми СRUD

12. Розробка клієнтської програми. Форми пошуку даних на клієнті.  (див. Лабораторну роботу №23) Включення до пояснювальної записки.

Реалізація коду форми пошуку даних на клієнті (apartmentSearchForm.cs):

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace EnergyConsumption

{

public partial class apartmentSearchForm : Form

{

private SqlConnection cnn = new SqlConnection();

private DataTable dt2 = new DataTable();

private BindingSource bs = new BindingSource();

public apartmentSearchForm()

{

InitializeComponent();

}

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

{

try

{

cnn.ConnectionString = "Data Source = (local); Initial Catalog = EnergyConsumption; Integrated Security = true";

cnn.Open();

SqlCommand cmd = new SqlCommand(

"SELECT FlatRegistry.FlatID, HeatingEquipment.Type " +

"FROM FlatRegistry " +

"INNER JOIN BuildingRegistry ON BuildingRegistry.HouseID = FlatRegistry.HouseID " +

"INNER JOIN HeatingEquipment ON HeatingEquipment.HeatingSystemID = BuildingRegistry.HeatingSystemID " +

"ORDER BY FlatRegistry.FlatID", cnn);

SqlDataAdapter da = new SqlDataAdapter(cmd);

int rowCount = da.Fill(dt2);

if (rowCount > 0)

{

bs.DataSource = dt2;

selectlistBox.DataSource = bs;

selectlistBox.DisplayMember = dt2.Columns[0].ColumnName;

selectlistBox.ValueMember = dt2.Columns[1].ColumnName;

}

else

{

MessageBox.Show("Данные не найдены.", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка: {ex.Message}", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

if (cnn.State == ConnectionState.Open)

cnn.Close();

}

}

private void listBox\_DoubleClick(object sender, EventArgs e)

{

try

{

if (selectlistBox.SelectedItem == null)

{

MessageBox.Show("Выберите квартиру из списка.", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

SqlCommand cmd = new SqlCommand("searchFlat", cnn);

cmd.CommandType = CommandType.StoredProcedure;

DataRowView dr = (DataRowView)selectlistBox.SelectedItem;

cmd.Parameters.Add("@search\_Flat", SqlDbType.NVarChar).Value = dr.Row.ItemArray[0].ToString();

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataTable dt = new DataTable();

da.Fill(dt);

if (dt.Rows.Count > 0)

{

string heatingType = dt.Rows[0]["HeatingSystemType"].ToString().Trim();

string flatId = dr.Row.ItemArray[0].ToString();

label4.Text = $"HeatingType: {heatingType} FlatID: {flatId}";

GridView.DataSource = dt;

}

else

{

MessageBox.Show("Данные для выбранной квартиры не найдены.", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка: {ex.Message}", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void label3\_Click(object sender, EventArgs e)

{

}

}

}

Реалізація форми пошуку даних на клієнті в батьківській таблиці на рисунку 6.

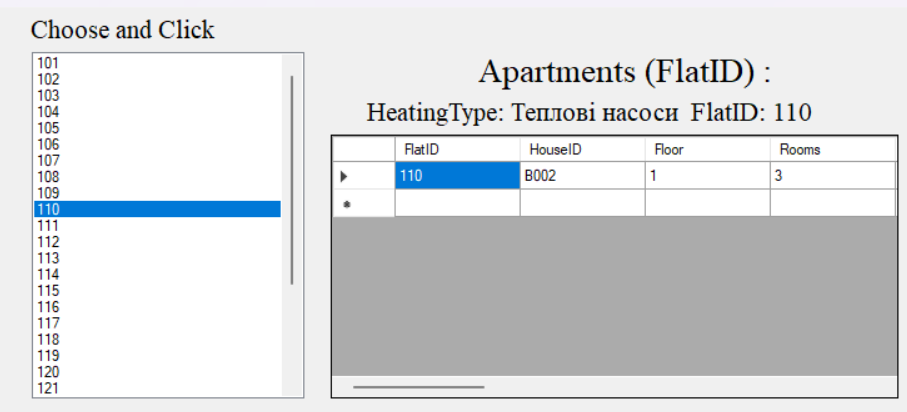


Рисунок 6 – Реалізація функціоналу форми пошуку даних

13. Розробка клієнтської програми. Форми експорту даних у документ Word (див. Лабораторну роботу №24) Включення до пояснювальної записки.

Реалізація коду клієнстької програми, що виконує форми експорту даних у документ Word (exportForm.cs):

using System;

using System.Data;

using System.Data.SqlClient;

using System.Windows.Forms;

using word = Microsoft.Office.Interop.Word;

namespace EnergyConsumption

{

public partial class exportForm : Form

{

SqlCommand cmd = new SqlCommand();

SqlConnection cnn = new SqlConnection();

SqlDataAdapter da = new SqlDataAdapter();

System.Data.DataTable dt = new System.Data.DataTable();

public exportForm()

{

InitializeComponent();

InitializeDatabaseConnection();

}

private void InitializeDatabaseConnection()

{

cnn.ConnectionString = "Data Source=(local);Initial Catalog=EnergyConsumption;Integrated Security=True";

cmd.Connection = cnn;

da.SelectCommand = cmd;

}

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

{

comboBox1.Items.Add("Вправа 3 Завдання 1");

comboBox1.Items.Add("Вправа 3 Завдання 2");

comboBox1.Items.Add("Вправа 3 Завдання 3");

comboBox1.Items.Add("Вправа 4 Завдання 1");

comboBox1.Items.Add("Вправа 4 Завдання 2");

comboBox1.Items.Add("Вправа 4 Завдання 3");

comboBox1.Items.Add("Вправа 5 Завдання 1");

comboBox1.Items.Add("Вправа 5 Завдання 2");

comboBox1.Items.Add("Вправа 5 Завдання 3");

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

dt.Rows.Clear();

dt.Columns.Clear();

switch (comboBox1.Text)

{

case "Вправа 3 Завдання 1":

textBox1.Text = "Виведення адрес даних про будинки, що знаходяться в Малиновському районі.";

cmd.CommandText = "SELECT \* FROM LocationAddresses WHERE District LIKE '%Малиновський%';";

break;

case "Вправа 3 Завдання 2":

textBox1.Text = "Виведення усіх даних про будинки, збудовані після 2000 року.";

cmd.CommandText = "SELECT \* FROM BuildingRegistry WHERE Year > 2000;";

break;

case "Вправа 3 Завдання 3":

textBox1.Text = "Отримати дані про будинки в яких є ліфт.";

cmd.CommandText = "SELECT HouseID, Year, Material, EnergyRating FROM BuildingRegistry WHERE Elevator=1;";

break;

case "Вправа 4 Завдання 1":

textBox1.Text = "Показати кількість даних квартир на кожному поверсі будинку.";

cmd.CommandText = "SELECT HouseID, Floor, COUNT(FlatID) AS NumberOfFlats FROM FlatRegistry GROUP BY HouseID, Floor ORDER BY HouseID;";

break;

case "Вправа 4 Завдання 2":

textBox1.Text = "Вивести інформацію про кількість квартир за кількістю мешканців, які в ній проживають.";

cmd.CommandText = "SELECT Residents, COUNT(FlatID) AS NumberOfFlats FROM FlatRegistry WHERE Residents IS NOT NULL GROUP BY Residents ORDER BY Residents;";

break;

case "Вправа 4 Завдання 3":

textBox1.Text = "Підрахувати загальні витрати на енергоресурси для кожної квартири.";

cmd.CommandText = "SELECT FlatID, Rooms, Area, Residents, TotalCost FROM FlatsTotalCostView;";

break;

case "Вправа 5 Завдання 1":

textBox1.Text = "Додати до завдання 4.3 інформацію про тип системи опалення для кожної квартири.";

cmd.CommandText = "SELECT FlatID, Rooms, Area, Residents, TotalCost, HeatingSystemType FROM FlatsTotalCostView;";

break;

case "Вправа 5 Завдання 2":

textBox1.Text = "Отримати список всіх будинків разом з даними про адресу та про систему отоплення.";

cmd.CommandText = "SELECT BuildingRegistry.HouseID, LocationAddresses.Address, LocationAddresses.District, HeatingEquipment.Type AS HeatingSystemType, BuildingRegistry.EnergyRating, BuildingRegistry.Year, BuildingRegistry.Material FROM BuildingRegistry INNER JOIN LocationAddresses ON BuildingRegistry.HouseID = LocationAddresses.HouseID INNER JOIN HeatingEquipment ON BuildingRegistry.HeatingSystemID = HeatingEquipment.HeatingSystemID;";

break;

case "Вправа 5 Завдання 3":

textBox1.Text = "Вивести інформацію про квартири, де використовують центральну систему опалення.";

cmd.CommandText = "SELECT FlatID, Rooms, Area, Residents, HouseID, HeatingSystemType FROM CentralHeatingFlatsView;";

break;

default:

break;

}

try

{

cnn.Open();

dt.Clear();

da.Fill(dt);

dataGridView1.DataSource = dt;

}

catch (Exception ex)

{

MessageBox.Show("Error: " + ex.Message);

}

finally

{

if (cnn.State == ConnectionState.Open)

cnn.Close();

}

}

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

{

if (comboBox1.Text != "Вправа 5 Завдання 2")

{

MessageBox.Show("Оберіть <Вправу 5 Завдання 2>");

return;

}

else

{

PrintToWord();

}

}

private void PrintToWord()

{

Microsoft.Office.Interop.Word.Application app = new Microsoft.Office.Interop.Word.Application();

Microsoft.Office.Interop.Word.Document doc = null;

string pathToExe = Application.StartupPath;

doc = app.Documents.Open(pathToExe + "\\sampleTask5\_2.docx");

doc.Activate();

Microsoft.Office.Interop.Word.Words wrds = doc.Words;

Microsoft.Office.Interop.Word.Range wRange;

wRange = wrds[11];

wRange.InsertAfter(DateTime.Today.Date.ToString("d"));

Microsoft.Office.Interop.Word.Bookmarks wBookmarks = doc.Bookmarks;

Microsoft.Office.Interop.Word.Bookmark mark;

mark = wBookmarks["kod\_House"];

wRange = mark.Range;

wRange.Text = dataGridView1.CurrentRow.Cells[0].Value.ToString().Trim();

mark = wBookmarks["address\_House"];

wRange = mark.Range;

wRange.Text = dataGridView1.CurrentRow.Cells[1].Value.ToString().Trim();

mark = wBookmarks["kod1\_House"];

wRange = mark.Range;

wRange.Text = dataGridView1.CurrentRow.Cells[0].Value.ToString().Trim();

mark = wBookmarks["fillTable"];

wRange = mark.Range;

wRange.Text = dataGridView1.CurrentRow.Cells[2].Value.ToString();

Microsoft.Office.Interop.Word.Table tab1 = doc.Tables[1];

for (int i = 2; i < 6; ++i)

{

wRange = tab1.Cell(2, i).Range;

wRange.Text = dataGridView1.CurrentRow.Cells[i + 1].Value.ToString();

}

app.Visible = true;

}

}

}

Реалізація виводу всіх запитів с 3 до 5-ої вправи на рисунку 7

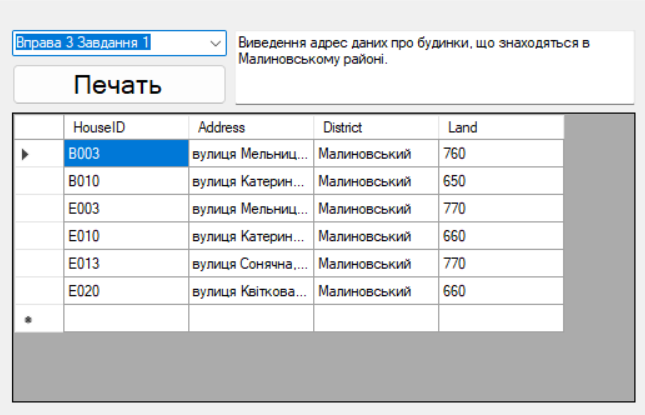


Рисунок 7 – Реалізація виводу даних запиту

Реалізація експорту вправи 5 запиту №2 у документ на рисунку 8.

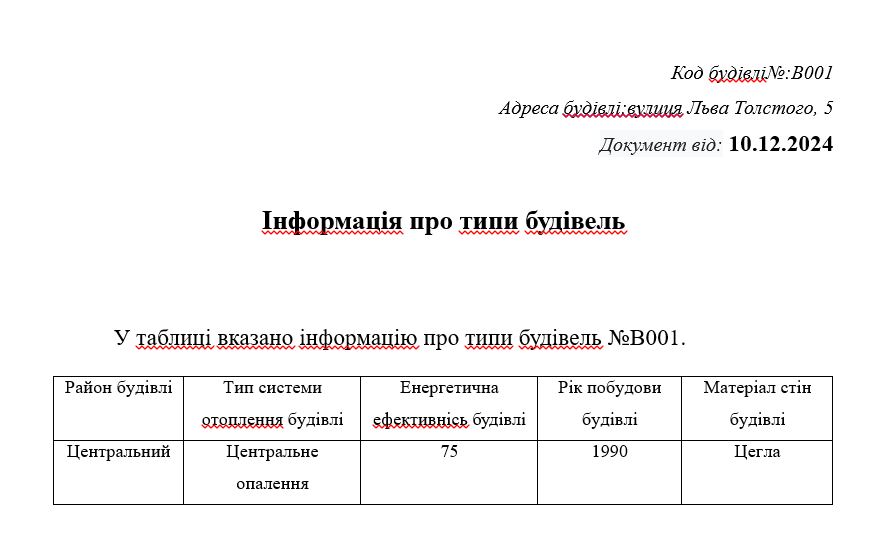


Рисунок 8 – Eкспорт даних у Word

# ВИСНОВОК

У ході виконання курсової роботи було спроектовано та реалізовано базу даних і клієнтську програму для обліку та аналізу енергоспоживання в багатоквартирних будинках. Вирішення поставлених завдань забезпечило створення інформаційної системи, що дозволяє здійснювати моніторинг і оптимізацію енергоспоживання у відповідності до чинного законодавства.

На основі проведеного аналізу предметної області було визначено ключові бізнес-функції системи та спроектовано функціональну модель, яка включає схему бази даних. Розроблена база даних містить всі необхідні таблиці для зберігання даних про об'єкти нерухомості, показники енергоспоживання, користувачів системи та інші супутні сутності.

Для організації доступу до даних було створено SQL-запити для виконання пошуку, підрахунку підсумків, об'єднання даних із кількох таблиць, а також модифікації та видалення даних. Використання уявлень, табличних виразів, збережених процедур і тригерів забезпечило гнучкість та автоматизацію обробки даних.

Клієнтська програма, розроблена у межах завдання, включає форми CRUD, які дозволяють управляти даними, форми пошуку, а також функціонал експорту інформації у документи Word. Завдяки цьому програмне забезпечення є зручним для користувача і відповідає вимогам функціональності.

Таким чином, розроблений програмний комплекс відповідає вимогам завдання, забезпечуючи ефективний моніторинг і аналіз енергоспоживання у багатоквартирних будинках, що сприятиме підвищенню енергоефективності та оптимізації витрат на енергоносії.

# ВИКОРИСТАНА ЛІТЕРАТУРА

1.Atkinson P., Vieira R. Beginning Microsoft SQL Server 2012 Programming. - Wiley, 2016, – 864с. ([Електронний ресурс] - Режим доступу: https://www.pdfdrive.com/beginning-microsoft-sql-server-2012-programming-e164606534.html)

2. Petkovic D. Microsoft SQL Server 2012: A Beginner's Guide.— McGraw-Hill, 2012. — 795 с. ([Електронний ресурс] - Режим доступу:  https://www.pdfdrive.com/microsoft-sql-server-2012-a-beginners-guide-e185318281.html).

3. Date C.J. SQL and Relational Theory: How to Write Accurate SQL Code. O’Reilly Media, Inc., 2015. — 584 с. ([Електронний ресурс] - Режим доступу: https://www.pdfdrive.com/sql-and-relational-theory-how-to-write-accurate-sql-code-e176192521.html)

4. Troelsen A.,  Japikse P. Pro C# 7: With .NET and .NET Core. - APress. Inc.,2017.- 1372 c. ([Електронний ресурс] - Режим доступу: https://www.pdfdrive.com/pro-c-7-with-net-and-net-core-d183552783.html)

5. Noyes B. Data Binding with Windows Forms 2.0: Programming Smart Client Data Applications with .NET. - Addison Wesley Professional, 2006. - 675с. ([Електронний ресурс] - Режим доступу: https://www.pdfdrive.com/data-binding-with-windows-forms-20-programming-smart-client-data-applications-with-net-e185459891.html)