МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Л.П. Згуровська, Ю.В. Киричук, Н.М. Назаренко

# БАЗИ ДАНИХ

## Комп'ютерний практикум

*Рекомендовано Вченою радою КПІ ім. Ігоря Сікорського*
*як навчальний посібник для здобувачів ступеня бакалавра, які навчаються за освітньою програмою «Комп'ютерно-інтегровані технології проектування приладів»*
*спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології»*

Київ
КПІ ім. Ігоря Сікорського
2021

Рецензенти: *Павловський О.М.,* канд. техн. наук, доцент кафедри комп`ютерно-інтегрованих оптичних навігаційних систем ї КПІ ім. Ігоря Сікорського

*Стахова А.П.* канд. техн. наук, доцент, доцент кафедри комп`ютерізованих електро-технічних систем та технологій Національного авіаційного університету

Відповідальний редактор: *Черепанська І.Ю.* докт. техн. наук, професор кафедри автоматизації та систем неруйнівного контролю КПІ ім. Ігоря Сікорського

*Гриф надано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 2 від 09.12.2021 р.) за поданням Вченої ради Приладобудівного факультету (протокол № 10/21 від 22.11.2021 р.)*

Електронне мережне навчальне видання

*Згуровська Людмила Петрівна***,** канд. техн. наук, доцент
*Киричук Юрій Володимирович***,** докт. техн. наук, доцент
*Назаренко Наталія Миколаївна*, асист.

# Бази даних.
# Комп'ютерний практикум

Згуровська Л.П. Бази даних. Комп'ютерний практикум [Електронний ресурс]: навч. посіб. для студ. спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології», освітньо-професійної програми «Комп'ютерно-інтегровані системи та технології в приладобудуванні» / Л.П. Згуровська, Ю.В. Киричук, Н.М. Назаренко; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 1,52 Мбайт). – Київ: КПІ ім. Ігоря Сікорського, 2021. – 241 с.

Основою навчального посібника є матеріали курсу лекцій «Бази даних», який викладається студентам приладобудівного факультету КПІ ім. Ігоря Сікорського. В навчальному посібнику розглянуто основи створення бази даних за допомогою СУБД Access. Навчальний посібник містить змістовний матеріал з теоретичних основ організації баз даних і систем керування базами даних. У посібнику розглянуто: загальні відомості про бази даних і системи керування базами даних, основні теоретичні поняття та терміни, які розкривають поняття баз даних, основи та принципи проектування баз даних, прийоми та засоби створення таблиць, простих та складних запитів, підлеглих форм і звітів та використання макросів для розв'язку задач.

Наведені комп'ютерні практикуми сприяють кращому засвоєнню матеріалу та отриманню навичок в реалізації баз даних у системі Microsoft Access. Задля закріплення та перевірки набутих навичок створеннята опрацювання баз даних наведено питання для самоконтролю з кожного розділу.

Для студентів технічних спеціальностей КПІ ім. Ігоря Сікорського.

**ЗМІСТ**

# Передмова

Навчальний посібник відповідає програмі дисципліни «Бази даних» для приладобудівного факультету Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» та призначений для студентів інженерно-технічних спеціальностей.

Головною метою даного навчального посібника є ознайомлення студентів із базами даних, методологією проектування баз даних, системами управління базами даних (СУБД), та здатністю застосовувати отримані знання для створення баз даних, використовуючи MS Access і MS Visual Studio, що читається для студентів приладобудівного факультету, обсягом теоретичного та практичного матеріалу, необхідного для засвоєння цього курсу.

Посібник складається з конспекту лекцій і комп'ютерних практикумів з дисципліни "Бази даних", яка традиційно викладається на перших курсах усіх технічних, економічних та соціологічних підрозділів Київського політехнічного інституту імені Ігоря Сікорського.

Слід відмітити, набуті знання після вивчення курсу дисципліни "Бази даних" ніякою мірою не є більш простою чи менш важливою у розумінні майбутнього практичного застосування. Навпаки, в результаті вивчення дисципліни «Бази даних» студенти отримують *знання*:

- основних принципів побудови нормалізованих баз даних;
- методів створення інтерфейсів для отримання довідкової і аналітичної інформації з бази даних;
- умов експлуатації та методів захисту від несанкціонованого доступу створеної бази даних;
- правил оформлення супровідної документації для створеної бази даних.

Студенти набудуть *уміння*:

- в отриманні необхідної науково-технічної інформації на етапі підготовки до створення бази даних відповідної предметної області, вивчення та критичного осмислення вже існуючих баз для розв'язку подібних задач;
- в формулюванні нових ідей по створенню баз даних;

- в обґрунтовуванні вибору системи управління базами даних та алгоритмічної мови програмування з урахуванням конкретних умов експлуатації; користуватися сучасними системами управління базами даних.

*Навички:*

- постановки задачі в області проектування бази даних та її практичної реалізації з використанням сучасних технологій;
- створення бази даних елементів приладів;
- створення бази даних характеристик елементів приладів;
- підбору елементів приладів на основі аналізу характеристик з використанням створеної бази даних.

*Досвід:*

- застосування методів аналізу та способів збору інформації предметної області, які необхідні для постановки задачі по створенню бази даних, методи нормалізації даних з метою отримання цілісності бази даних;
- проведення експериментальних досліджень основних характеристик створеної бази, оформлення супровідної документації для створеної бази даних .

Посібник може бути рекомендовано студентам другого курсу для більш ефективного засвоєння лекційних матеріалів, а також стати основою дистанційного курсу із зазначених розділів, що допоможе студентам всіх форм навчання або старшокурсникам при спеціалізації як довідниковий матеріал.

При написанні навчального посібника використано багаторічний досвід викладання бази даних студентам приладобудівного факультету Київського політехнічного інституту імені Ігоря Сікорського.

Матеріал посібника розбито на дванадцять лекцій. Зміст розглянутих в лекціях питань відповідає змісту навчального посібника. Теоретичний матеріал ілюструється достатньою кількістю прикладів, що дозволяє студентам самостійно розібратися в даній темі. Наприкінці кожного розділу посібника запропоновано добірки індивідуальних завдань, які можуть бути використані студентами для закріплення викладеного матеріалу, а також для проведення викладачами різних видів контролю знань студентів.

Даний навчальний посібник допоможе більш глибокому засвоєнню викладених розділів та сприятиме подальшому вивченню бази даних.

# Комп'ютерний практикум №1. Створення бази даних в СУБД Access

*Мета роботи* - набути навичок створення бази даних за допомогою СУБД Access. Вивчити основні об’єкти бази даних: типи даних, таблиці та їх властивості.

## Теоретичні відомості

Access це система управління реляційними базами даних (СУБД). Середовище СУБД Access складається з таких об’єктів : таблиці, форми, звіти, запити, сторінки, макроси, модулі. Не всі з перелічених об’єктів обов’язково повинні входити до бази даних і залежить це від вимог до створюваної бази даних. Форми та звіти можуть використовувати елементи керування, перелік яких можна побачити на відповідній панелі при створенні згаданих об’єктів. До елементів керування відносяться кнопки; текстові поля різних форматів: прості поля, списки, поля зі списками, поля, значення яких обчислюються; надписи, які використовуються для опису вмісту відповідного елемента керування чи об’єкта; перемикачі та групи перемикачів і т.і. До речі, текстові поля можуть бути як зв’язаними, так вільними. Зв’язані поля мають зв’язок з полями відповідної таблиці і тому, використовуючи форму чи звіт, значення цих полів можна переглядати, змінювати, видаляти. Вільні поля використовуються лише для виведення даних у форму чи звіт. Це можуть бути поля, значення яких обчислюються на основі даних полів таблиці, наприклад, сума придбаного товару або використовуватися для виведення службової інформації, наприклад, виведення поточної дати. Кожен з об’єктів має визначений список атрибутів, що називаються властивостями, за допомогою яких можна встановлювати зовнішній вигляд та поведінку даного об’єкта. Щоб переглянути властивості об’єкта, необхідно виділити його та натиснути праву кнопку миші і вибрати з контекстного меню Свойства відповідного об’єкта, наприклад, Свойства таблицы. Крім властивостей об’єкта Таблиця, в контекстному меню розташовані команди, які можна виконувати при роботі з таблицями (рис. 1.1).

Середовище СУБД Access надає універсальні можливості для роботи з даними, які об’єднані у таблиці. Дані в таблицях можна об’єднувати за змістом

(дані про співробітників, перелік матеріалів на складі, відомість для отримання стипендії і т.і.). Таблиця даних має наступну структуру: рядки та стовпчики. Рядки таблиці отримали назву – записи(кортежі), а стовпчики – поля (атрибути). У стовпчиках таблиці розміщується тільки дані одного типу і однакового змісту( наприклад, у одному стовпчику таблиці **Лічильники** знаходиться назва обладнання, у другому – діаметр умовного проходу і т.і.).

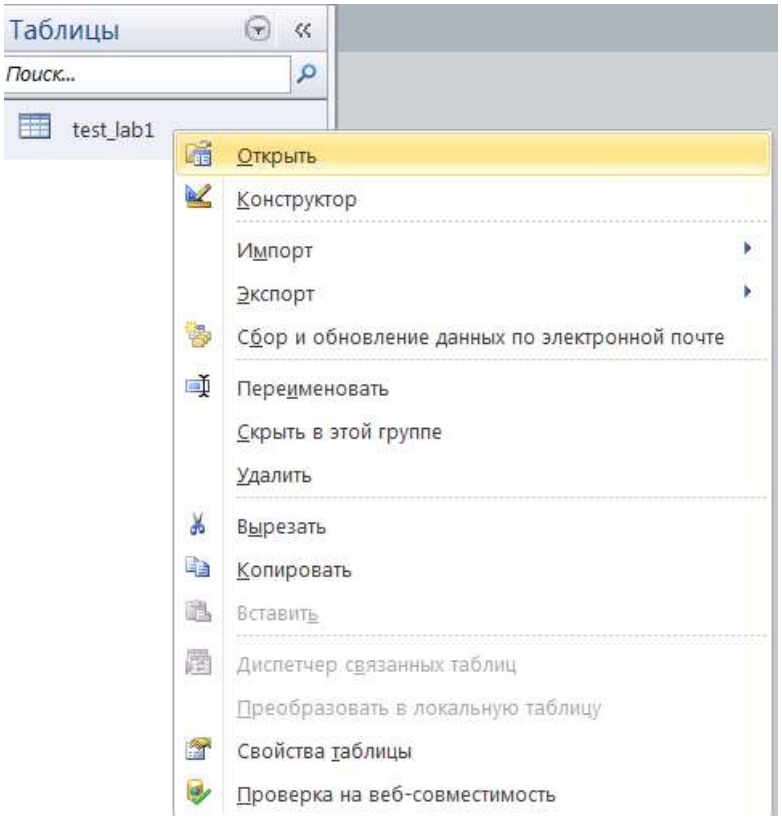



Рис. 1.1. Контекстне меню об’єкта таблиця СУБД Access

## 1.1. Створення таблиць за допомогою конструктора

Для створення нової бази даних необхідно викликати Microsoft Access. Запуск СУБД Access супроводжується відкриттям відповідного вікна, в якому для створення нової бази даних потрібно вибрати шаблон **Новая база данных** (рис. 1.2,а), після чого у вікні (рис. 1.2,б), потрібно набрати ім’я створюваної бази даних, наприклад, FFF_db_var_N_DDDD, де FFF- прізвище студента, var_N – номер варіанта індивідуального завдання, DDDD – рік виконання комп’ютерного практикуму та вибрати раніше створену папку FFF_NNNN (FFF

– це прізвище студента, NNNN -шифр групи), в якій ця база даних буде зберігатися і натиснути кнопку **Создать.**

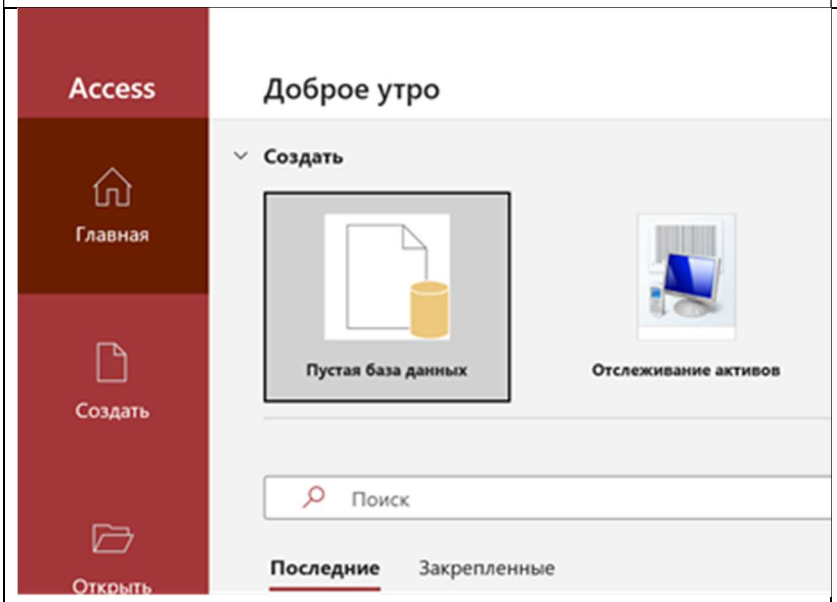
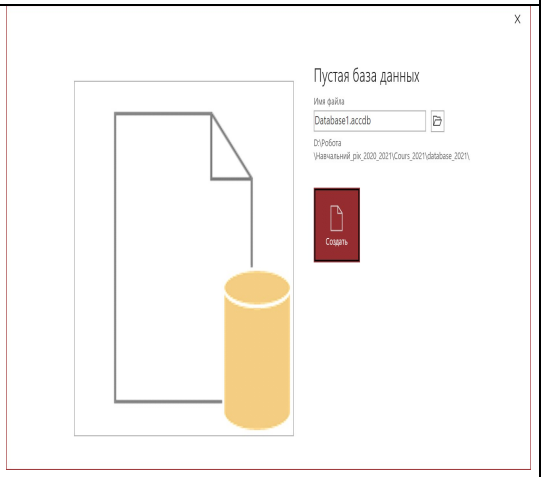
| При запуску СУБД Access 2019 на виконання, відкривається вікно з доступними шаблонами. | Для створення нової бази вибрати відповідний шаблон та |
|---|---|
|  а) |  б) |

Рис. 1.2. Вікно запуску СУБД Access для створення нової бази даних

Ці дії необхідно виконати лише для створення нової бази даних. Процес збереження нової бази даних завершується відкриттям вікна створення нових об’єктів в СУБД Access . Це вікно ще називають скорочено – вікно бази даних СУБД Access. Нова база даних відкривається в режимі програми-майстра по створенню таблиці (рис. 1.3).

Для створення об’єкту таблиця можна використовувати інший спосіб. Для цього в головному меню СУБД Access треба вибрати закладку **Создание** і натиснути піктограму **Конструктор таблиц** у вікні бази даних (рис. 1.4).

Для того, щоб створити нову таблицю можна вибрати **Работа с таблицами**, розкрити список пікторами **Режим** і обрати **Конструктор** (рис. 1.5).

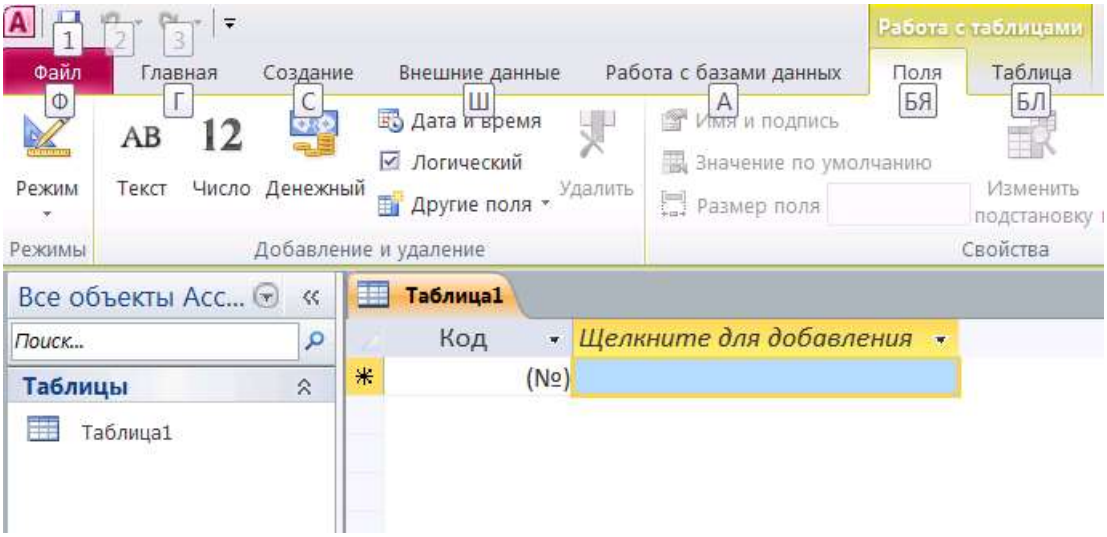



Рис. 1.3. Вікно конструктора створення структури таблиці бази даних

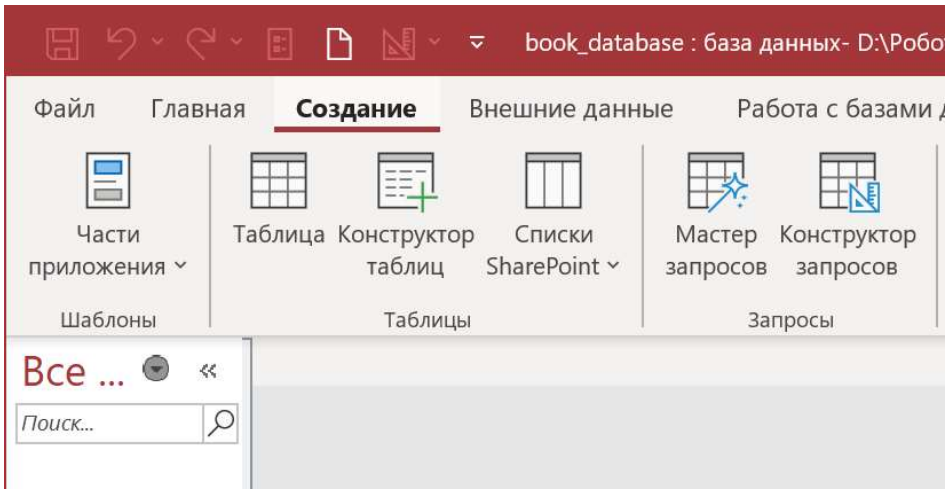



Рис. 1.4. Вікно створення нового об’єкта таблиця в СУБД Access

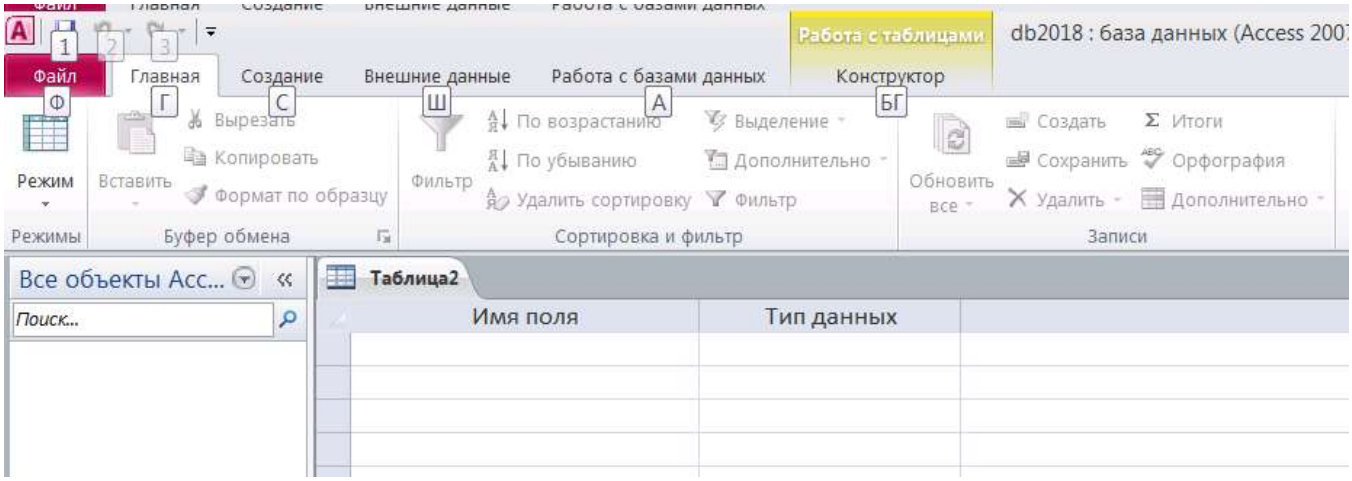



Рис. 1.5. Створення нової таблиці в режимі конструктора

Далі створити структуру нової таблиці (рис. 1.6). Обов'язково мають бути заповненими поля **Имя поля** та **Тип данных.** Заповнення поля **Описание** не є обов'язковим, але інформація цього поля буде корисною для подальшого супроводу бази даних іншими спеціалістами. В полі **Описание** записують розширену інформацію про дані, що зберігатимуться у полі **Имя поля.** Наприклад, **Имя поля –** NazvaP, а **Описание** – назва компанії - виробника. У таблиці 1 представлені всі доступні в СУБД Access типи даних. Тип даних та властивості кожного поля вибираються в залежності від вмісту даних, які потім будуть зберігатися у відповідному полі. Поля таблиць можуть належати одному з двох типів : прості поля та ключові поля.

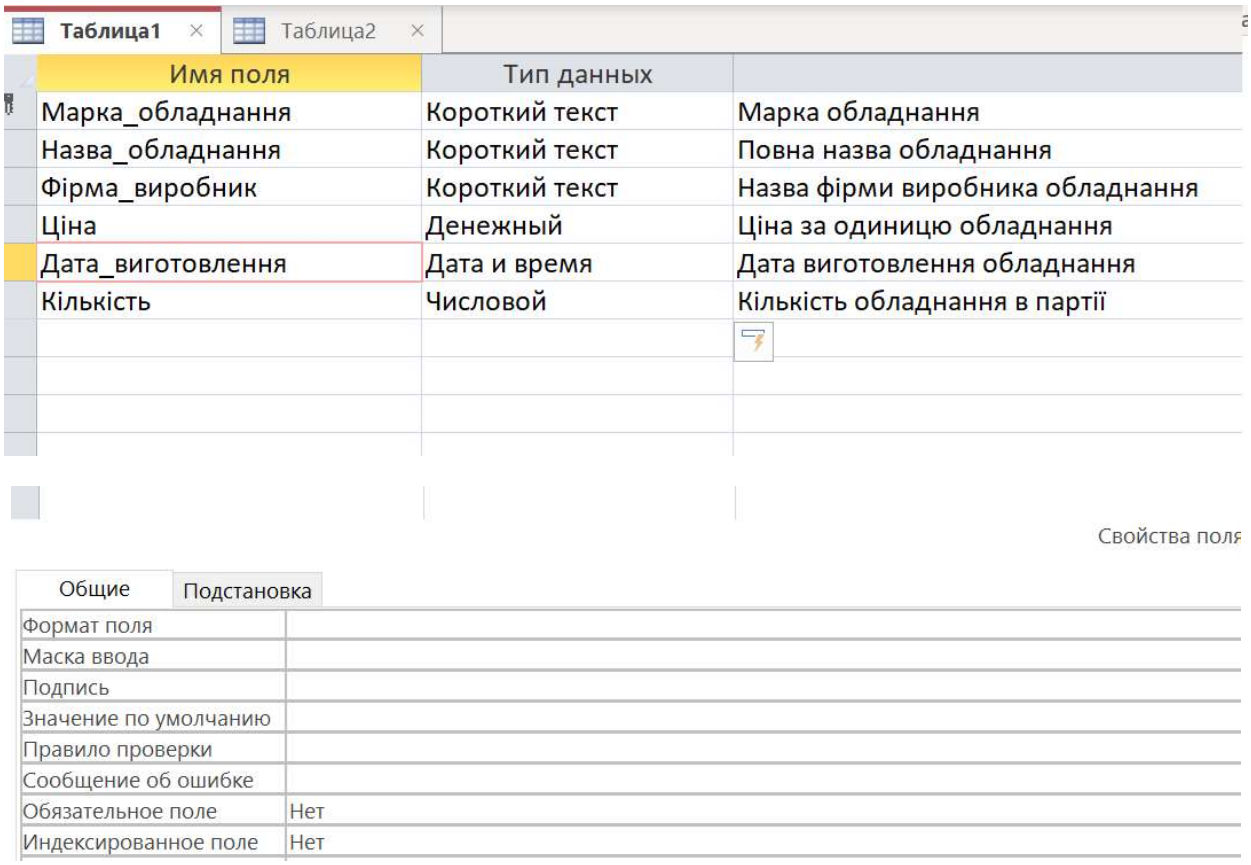



Рис. 1.6. Зразок структури новостворюваної таблиці з вікном властивостей

В простих полях зберігаються безпосередньо дані. Ключові поля крім збереження даних використовуються для організації зв'язків між таблицями. Тому вони називаються ключами. Ключ зазвичай складається з одного поля, але може складатись і з декількох полів.

Для того щоб створити складений ключ із кількох полів, необхідно клацнути мишею на маркері виділення рядка для кожного поля, утримуючи натиснутою клавішу **Ctrl.**

На вкладці **Конструктор** у групі **Поля** виберіть елемент **Первинний ключ**. Піктограма ключа додається ліворуч від поля або полів, вибраних як первинний ключ.

Існують два типи ключів:

*Первинний ключ.* Таблиця може мати лише один первинний ключ. Первинний ключ складається з одного або декількох полів, які унікально ідентифікують кожен запис, що зберігається в таблиці. Як первинний ключ часто використовують певний унікальний ідентифікатор, наприклад ідентифікаційний номер, серійний номер або код. Наприклад, в таблиці "Таблиця1", у якій кожне обладнання має унікальне значення поля **Марка_обладнання**. Таке поле– це первинний ключ таблиці "Таблиця1". Якщо первинний ключ містить кілька полів, зазвичай він складається з наявних полів, які, взяті разом, та забезпечують унікальні значення. Наприклад, можна використати поєднання поля **Код_обладнання** з полем **Діаметр_умовного_проходу**, як первинний ключ для таблиці **Лічильники**.

*Зовнішній ключ.* Таблиця також може мати один або кілька зовнішніх ключів. Зовнішній ключ містить значення, які відповідають значенням первинного ключа іншої таблиці [1].

Кожному типу даних (табл. 1) відповідає список властивостей (табл. 2), який відображається у нижній частині вікна конструктора таблиць (рис. 1.6) [3].

Таблиця 1. Типи даних СУБД Access

| Найменування типу даних | Зміст | Максимальне значення даного типу |
|---|---|---|
| Текстовый | (Значення за замовчуванням). Текст або числа, що не вимагають проведення розрахунків, наприклад, номери телефонів. | Число символів, що не перевищує мінімальне із двох значень: 255 або значення властивості Розмір поля (FieldSize). Microsoft Access не |

| | | |
|---|---|---|
| | | зберігає порожні символи в частині поля, що не використовується. |
| Поле MEMO | Довгий текст або комбінація тексту й чисел. | До 65535 символів. (Якщо поле MEMO обробляється через об'єкти доступу до даних (DAO) і містить тільки текст і числа, а не двійкові дані, то його розмір обмежується розміром бази даних). |
| Числовой | Числові дані, що використовуються для проведення розрахунків. | 1,2,4 або 8 байт (16 байт тільки для коду реплікації). |
| Дата/время | Дата і час, що відносяться до років з 100 по 9999 включно. | 8 байт. |
| Денежный | Грошові значення й числові дані, що використовуються в математичних розрахунках, що проводяться з точністю до 15 знаків у цілій і до 4 знаків у дробовій частині. | 8 байт. |
| Счетчик | Унікальні послідовно зростаючі (на 1) або випадкові числа, що автоматично вводяться при додаванні кожного нового запису в таблицю. Значення полів типу лічильника обновляти не можна. | 4 байт (16 байт, якщо для властивості Розмір поля (FieldSize) задане значення коду реплікації). |
| Логический | Логічні значення, а також поля, які можуть містити одне із двох можливих значень (True/False, Так/Ні). | 1 біт. |
| Поле объекта OLE | Об'єкт (наприклад, електронна таблиця Microsoft Excel, документ Microsoft Word, малюнок, звукозапис або інші дані у двійковому форматі), зв'язаний або впроваджений у | До 1 Гбайт (обмежується обсягом диска). |

| | | |
|---|---|---|
| | таблицю Microsoft Access. | |
| Гиперссылка | Рядок, що складається з літер і цифр і представляє адресу гіперпосилання. Адреса гіперпосилання може складатися максимум із трьох частин: текст s- текст, виведений у поле або в елементі керування; адреса s- шлях до файлу (у форматі шляху UNC) або сторінці (адреса URL); додаткова адреса s- зсув усередині файлу або сторінки. | Кожна із трьох частин у типі Гіперпосилання може містити до 2048 символів. |
| Мастер подстановок | Створює поле, у якому пропонується вибір значень зі списку або з поля зі списком, що містить набір постійних значень або значень із іншої таблиці. Вибір цього параметра в списку поля запускає майстра підстановок, що визначає тип поля | Той же розмір, що й ключового поля, що використовується в підстановці (звичайно 4 байт). |

Для кожного створюваного поля таблиці існує список властивостей. Значення властивостей можна залишити тими, що пропонує СУБД Access або задавати нові значення відповідно до вимог поставленої задачі. Опис властивостей можна знайти в довідковій системі СУБД Access та таблиці 2.

Таблиця 2. Список властивостей полів таблиці

| Назва властивості | Призначення властивості |
|---|---|
| Размер поля (Field Size) | Значення цієї властивості вказує максимальний розмір даних, які можуть зберігатися в полях текстового (Text), числового(Number) типа і в полях типу счетчик (AutoNumber). Для текстового поля ця властивість може містити число від 0 до 255. Всі символи рядка, що зберігаються в полі і які виходять за зазначений розмір, відкидаються. Для поля типу счетчик (AutoNumber) допускаються лише два значення -длинное целое (Long Integer) і код репликації (Replicatin ID). |
| Формат поля (Format) | Значення цієї властивості управляє способом відображення даних, що зберігаються в полях всіх типів, крім поля объекта OLE (OLE Object). Формат поля ніяк не впливає на те, яким способом дані зберігаються в таблиці. |
| Точность (Presition) | Ця властивість доступна тільки для полів числового (Number) типу, у яких властивості Размер поля (Field Size) має значення Действительное (Decimal) і оз¬начає точність подання чисел. Значенням цієї властивості є загальна кількість цифр, що бере участь у поданні числа, як праворуч від десяткової крапки, так і ліворуч від її. Ця властивість задає ступінь деталізації числа, а отже – ступінь точності. Так, наприклад, ступінь деталізації числа 2.123456 більше, ніж ступінь деталізації числа 2.3. |
| Масштаб (Sca1e) | Ця властивість також доступна для числових полів, у яких властивість |
| Размер поля (Field Size) | встановлено значення Действительное (Decimal). Значенням цієї властивості є кількість знаків праворуч від десяткової крапки, які будуть зберігатися в таблиці. |

| | |
|---|---|
| Количество десятичных знаков (Decimal Places) | Ця властивість доступна для числовых (Number) і денежных (Currency) типів полів. Вказує кількість знаків, що відображаються праворуч від десяткової крапки (зрівняти з попередньою властивістю). Значення за замовчуванням — Авто (Auto), тобто число відображається так, як воно було введено. |
| Маска ввода (Input Mask) | Ця властивість призначена для полегшення контролю над введенням користувачем специфічної інформації в поле таблиці, запиту або в елемент керування форми або звіту. Типові приклади використання маски введення – паролі, телефонні номери, дата, час. Маски введення, як і більшість нових об'єктів Microsoft Access, можна створити двома різними шляхами – за допомогою майстра й вручну. Для запуску майстра потрібно натиснути кнопку для побудови (кнопка із трьома крапками праворуч від поля) властивості Маска ввода (Input Mask). Варто помітити, що майстер масок введення можна використати тільки для полів текстового (Text) типу і полів дата/время (Date/Time), незважаючи на те, що кнопка для побудови доступна й для числовых (Number), і для денежных (Currency) типів. |
| Подпись (Caption). | Властивість подпись є присутньою для полів всіх типів і може містити до 2048 символів текстової інформації. Якщо ця властивість містить який-небудь текст, то він буде використовуватися як заголовок стовпця, інакше в цій якості буде використане найменування поля. |

| | |
|---|---|
| Значение по умолчанию (Default Value). | Ця властивість властива полям всіх типів за винятком полів типу счетчик (AutoNumber) і объект OLE (OLE Object). У цій властивості вказується значення, що буде автоматично підставлено в поле при створенні нового запису. Можна вказати також і вираз. Наприклад, вираз «=Date()» повертає поточну системну дату. Знак «=» перед виразом потрібно вказувати обов'язково. Значення виразу буде обчислено на момент додавання нового запису й підставлено у відповідне поле. Ви можете ввести в це поле інше значення або залишити підставлене значення без зміни. Максимальна довжина цієї властивості – 255 символів. |
| Условие на значение (Validation Rule) і Сообщение об ошибке (Validation Text). | Ці властивості застосовні до полів всіх типів за винятком полів типу счетчик (AutoNumber) і объект OLE (OLE Object). |
| Обязательное поле (Required). | Цю властивість можна вказати для полів всіх типів, крім счетчика (AutoNumber), яке є обов'язковим по визначенню. Можливі значення – так (Yes) чи Ні (No). Якщо поле є обов'язковим, то в нього потрібно ввести підходящі дані — інакше буде видане повідомлення про помилку. Значення за замовчуванням — Ні (No). Якщо в полі введений пробіл, то він буде зігнорований, і поле буде вважатися порожнім (мати значення Null). |
| Пустые строки (Allow Zero Lenght). | Властива текстовым ( Text) полям, полям Мето і гиперссылкам (Hyperlink). З його допомогою можна дозволити або заборонити введення в поле порожніх рядків. Можливі значення - Так ( Yes) чи Ні (No). Значення за замовчуванням - Ні (No). Порожній рядок можна ввести, вказавши в поле пару подвійних лапок. |

| | |
|---|---|
| Индексированное поле (Indexed). | Властивість вказує, чи буде поле індексованим. З його допомогою можна задати простой индекс (що складається з одного поля) для текстовых (Text), числовых (Number); денежных (Currency), логических (Yes/No) полів, а також полів типу дата/время (Date/Tiте). Можливі значення - Ні (No), Так (допускаються збіг), Yes (Duplicates OK)) і Так (збіг не допускаються) ( Yes (No Duplicates)). |
| Сжатие Юникод (Uncode Compression). | Включає або відключає стиснення Юникод для текстовых ( Text) полів, полів Мемо і гиперссылок (Hyperlink). Починаючи з версії Microsoft Access 2000, текстова інформація в таблицях зберігається в кодуванні Юникод. Це означає, що на кожен символ приділяється не один байт, а два. Кодування Юникод підтримує до 65535 символів, що дозволяє використати символи різних національних алфавітів. З іншого боку, це веде до перевитрати дискового простору. Якщо стиснення включено, то всі символи, перший байт яких дорівнює нулю (по більшій частині це символи латинського алфавіту), будуть зтискуватись при записі на диск, а при вибірці - відновлюватися. За замовчуванням кодування Юникод включене. |
| Режим IME(IME Mode) і Режим предложений IME(IME Sentence Mode). Input Method Editor (IME) - | це механізм, що дозволяє вводити символи мов азіатської групи (спрощений або традиційний китайський, японський, корейський і т.д.) шляхом відповідного перетворення кодів натиснутих клавіш на клавіатурі. Значення за замовчуванням для властивості Режим IME (IME Mode) — Нет контроля (No Contro1), тобто перетворення не використовується. Для властивості Режим предложений IME (IME Sentence Mode) значення за замовчуванням - Ні (Nопе). |

| | |
|---|---|
| Новые значения (New Values). | Значення цієї властивості вказує, яким образом будуть підбиратися нові величини для поля типу счетчик (AutoNumber) при створенні нового запису. Можливі значення - Последовательные (Increment) і Случайные (Random). Значення за замовчуванням – Последовательные (Increment) . Це значення є найкращим вибором у переважній більшості випадків. |

Після завершення створення структури таблиці її необхідно зберегти з відповідним ім'ям. Для заповнення таблиці даними потрібно виділити ім'я збереженої таблиці і вибрати опцію **"Открыть"** у вікні бази даних і заповнити відкриту таблицю даними. Але доцільніше заповнювати таблиці даними, використовуючи інший об'єкт СУБД Access – форму.

Всі об'єкти СУБД Access , а саме: таблиці, форми, звіти, запити, сторінки, макроси, модулі створюються подібно до створення таблиць. Але деякі особливості створення інших об'єктів СУБД Access будуть розглянуті у наступних комп'ютерних практикумах.

В СУБД Access таблиці можна не тільки створювати , але й імпортувати або копіювати. Імпортувати таблиці можна з іншої бази даних Access. Копіювання таблиць відбувається в межах однієї бази даних.

## 1.2. Імпорт таблиць даних

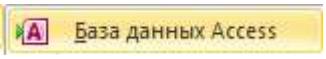
Для імпорту таблиць даних потрібно на вільному від піктограм таблиць місці вікна **Таблицы** необхідно натиснути правою кнопкою миші та вибрати пункт контекстного меню **Импорт**. З'явиться вікно **Импорт**, в якому потрібно вибрати 

рис. 1.7. Далі обрати **джерело і місце призначення даних** куди ми хочемо імпортувати дану таблицю.

Таблиці будуть вбудовані у вашу базу даних. Операцію імпортування можна застосовувати і до інших об'єктів бази даних.

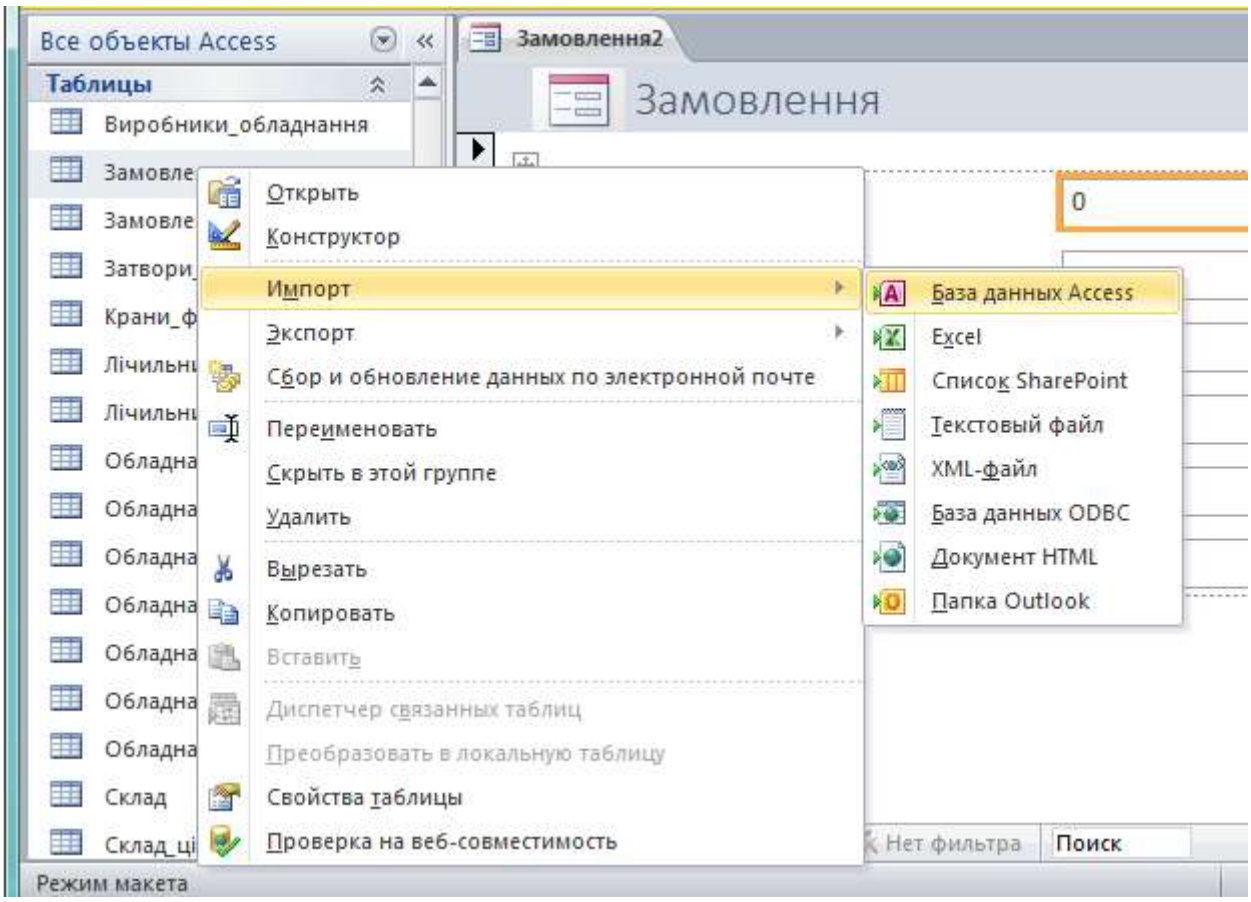



Рис. 1.7. Вікно опцій імпортування таблиць

## 1.3. Копіювання даних з таблиць

Виконуючи операцію копіювання даних з таблиці можна скопіювати лише структуру таблиці, або структуру і дані таблиці, або додати дані до таблиці . Процес копіювання складається з наступних кроків.

1. Відкрити базу даних з таблицями**.**
2. Натиснути правою кнопкою миші на таблиці. Наприклад, вибрати таблицю **Лічильники** та обрати в контекстному меню опцію **Копировать**.
3. На вільному від піктограм таблиць місці вікна бази даних натиснути праву кнопку миші та вибрати пункт контекстного меню **Вставить**.
4. У вікні **Вставка таблицы** у текстовому полі **Имя таблицы** ввести нову назву таблиці, наприклад, **Крани**, в яку будуть скопійовані структура і дані таблиці **Лічильники** та встановити перемикач **Параметры вставки: Структура и данные**.

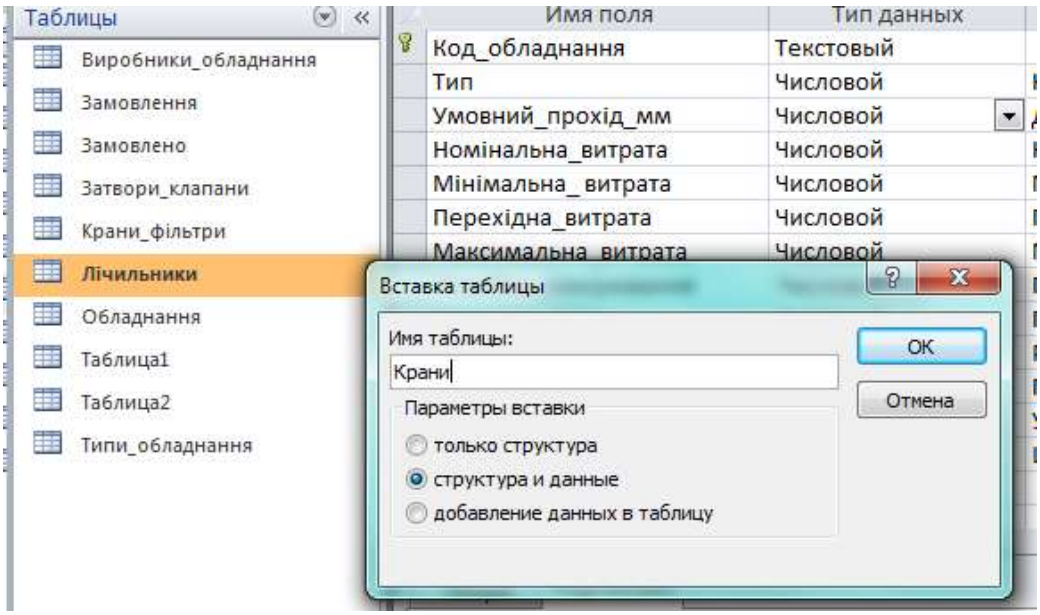



Рис. 1.8. Вікно опцій копіювання таблиць

Операцію копіювання можна застосовувати і до інших об’єктів бази даних.

*Використання* ***Мастер подстановок***

Розглянемо використання **Мастера подстановок** для поля Установка_лічильників таблиці “Лічильники. Відкриємо таблицю в режимі Конструктор і в стовпчику Тип даних оберемо **Мастер подстановок**.

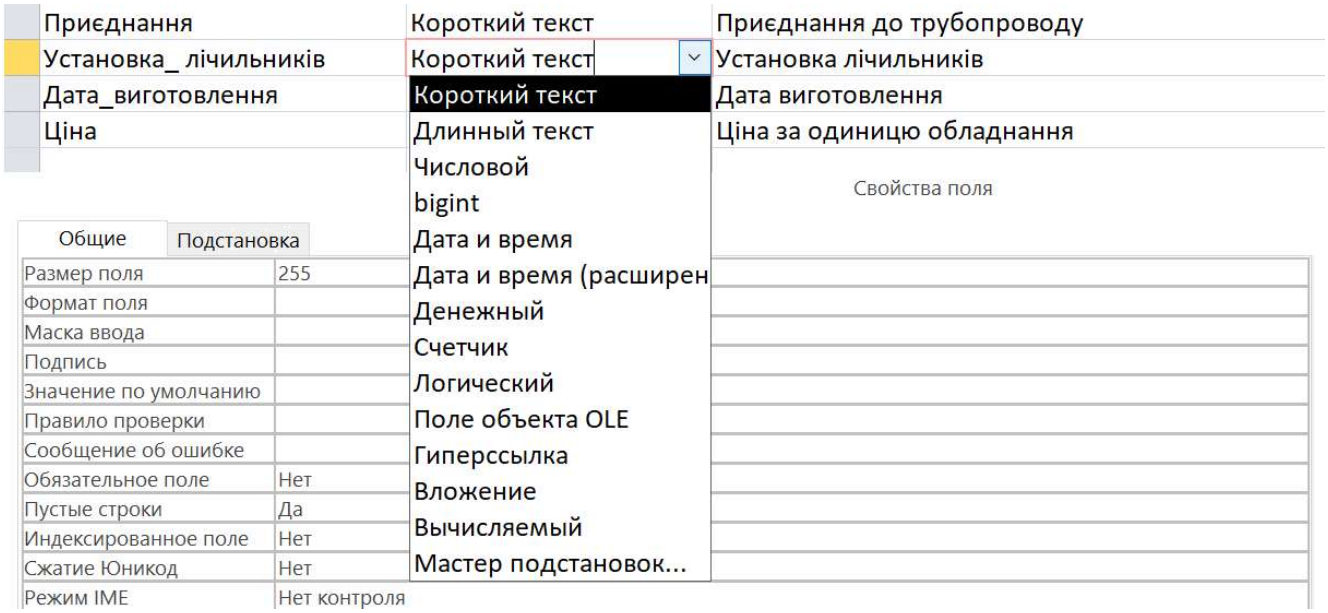



Рис. 1.9. Вибір Мастер подстановок

З’явиться вікно (рис. 1.10), в якому потрібно вибрати **Будет введен фиксированный набор значений**. Натиснути кнопку **Далее.**

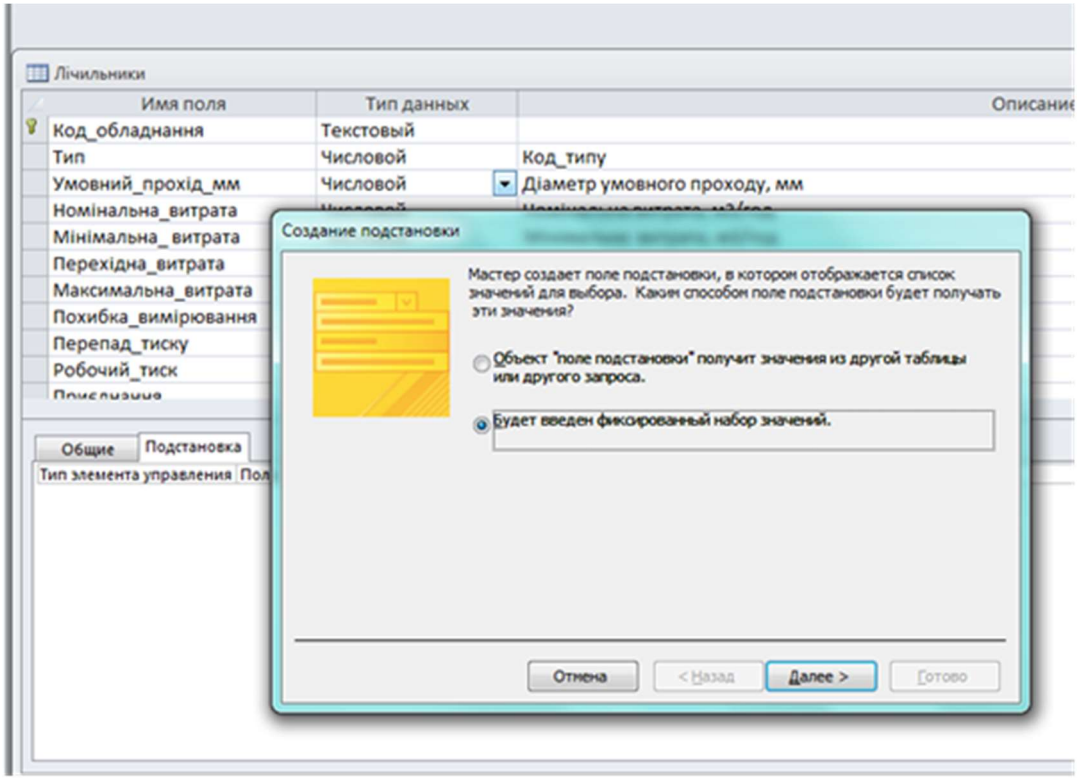



Рис. 1.10. Структура вікна **Мастер подстановок**

Для поля Установка_лічильників таблиці Лічильники ввести значення поля: горизонтально, вертикально, горизонтально та вертикально і натиснути “**Далее**”. Потім натиснути кнопку “**Готово**”.

*Зв’язки між таблицями в СУБД Access*

В застосуванні, що створюється в реляційної СУБД Access, кожен запис в будь-якій таблиці повинен унікально ідентифікуватися. Тобто значення деяких полів в таблиці не повинні повторюватися у всіх інших записах цієї таблиці. Цей унікальний ідентифікатор, який використовується як назва стовпчика, називається первинним ключем. Після завершення створення структури таблиці в режимі **Конструктор**, при виконанні операцій "Закрити" або "Зберегти", Access робить нагадування.

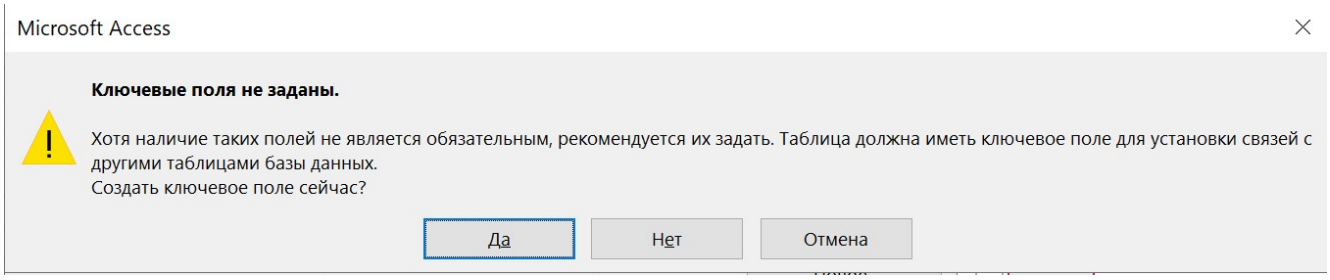



Рис. 1.11. Повідомлення з нагадуванням про необхідність створення первинного ключа

Мета правильної розробки таблиць – видалення надлишкових даних та підтримка цілістності даних. Для досягнення цієї мети таблицю розділяють на багато тематичних таблиць, щоб кожен факт було представлено лише один раз. Потім у програмі Access використовуються засоби, за допомогою яких ці розділені дані можна зібрати разом – для цього потрібно вставити в пов’язані між собою таблиці спільні поля, які будуть виконувати роль ключових. Проте для того, щоб зробити це належним чином, необхідно зрозуміти, які зв’язки існують між таблицями, а потім вказати ці зв’язки в базі даних [3].

Зв’язок в Access дає змогу об’єднати дані з двох різних таблиць. Кожний зв’язок складається з полів у двох таблицях із відповідними даними. Наприклад, таблиці Назва_обладнання та Склад_обладнання можуть мати однакове поле Код_типу. Кожен запис у таблиці " Склад_обладнання має код обладнання у відповідному полі, який відповідає запису в таблиці Назва_обладнання з таким самим кодом типу.

В базах даних існує три види зв’язків: один-до-одного, один-до-багатьох, багато-до-багатьох.

Зв'язок типу один-до-одного означає, що поле однієї таблиці пов'язано з одним полем іншої таблиці. Зв'язок один-до-одного найчастіше свідчить про те, що одна таблиця невірно розділена на дві окремі.

Зв'язок типу один-до-багатьох означає, що поле однієї таблиці пов'язано з декількома полями іншої таблиці. Це найбільш часто використовуваний тип зв'язку. Таблиця (з боку "один") називається батьківського, а таблиця (з боку "багато") – дочірньою. Характерний приклад такого зв'язку наведено на рис. 1.12, де таблиця Назва_обладнання є батьківського, а таблиця Склад_обладнання – дочірньою.

Зв'язок типу багато-до-багатьох означає, що кожне поле однієї таблиці може бути пов'язане з декількома полями іншої таблиці , і навпаки. Тип зв'язку багато-до-багатьох є тимчасовим типом зв'язку, допустимим на ранніх етапах розробки структури застосування. В подальшому цей тип зв'язку має бути замінений двома зв'язками типу один-до-багатьох шляхом створення додаткової таблиці.

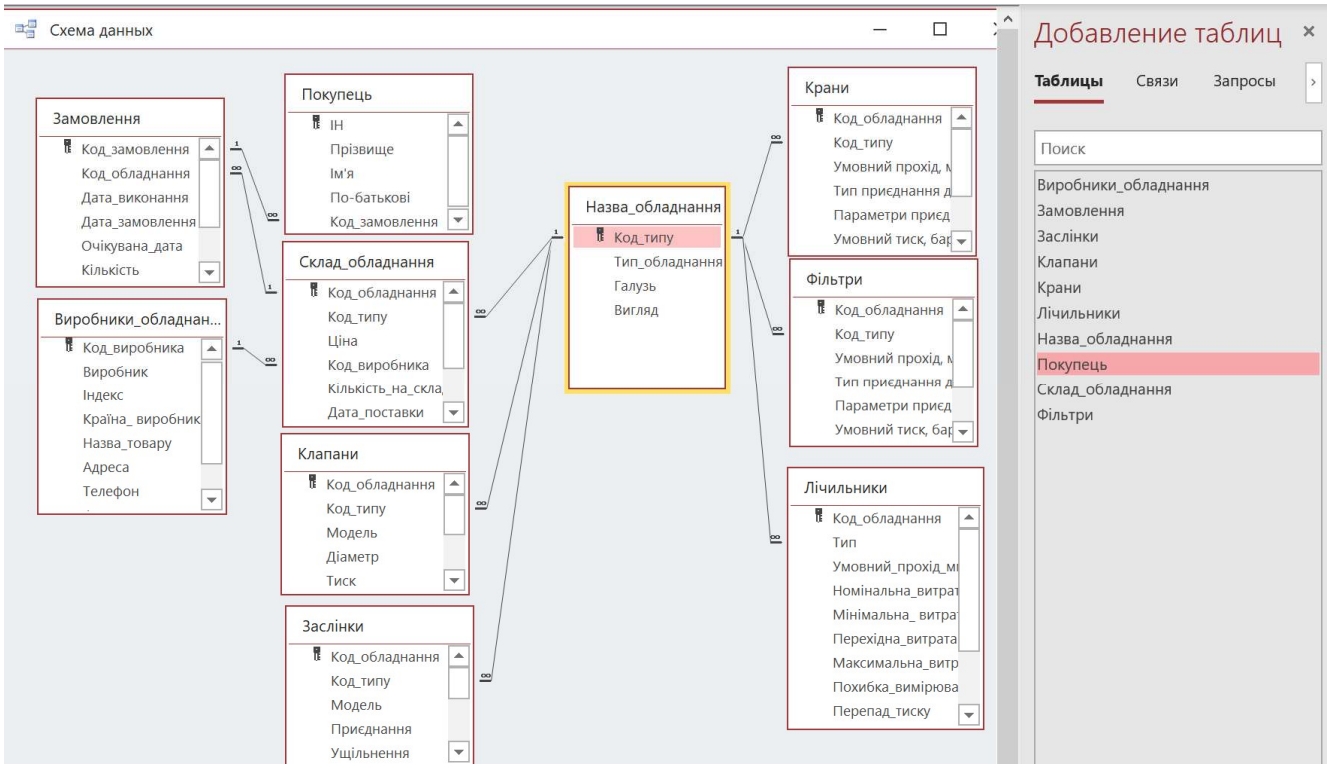



Рис. 1.12. Вікно **Схема данных** з встановленими зв’язками між таблицями бази даних db_stud_comp_prak.accdd

Використання зв’язків таблиць в об’єкті Запит, дає Access змогу визначити, які записи з кожної таблиці слід поєднувати в віртуальній таблиці результатів. Крім того, вірно встановлені зв’язки між таблицями допомагають запобігти втраті даних. Це називається цілісністю даних застосування Access [3].

*Створення, редагування та видалення зв’язків*

У версіях СУБД Access, починаючи з Access 2010, схема даних створюється автоматично. Але якщо в базі даних з’явилася нова таблиця, то використовуючи опцію головного меню

Работа с базами данных та вибору піктограми Схема данных, можна встановити зв’язок між таблицями. З контекстного меню вікна Схема данных

необхідно вибрати опцію Добавить таблицу, перетягнути потрібну таблицю з вікна із списком таблиць бази даних.

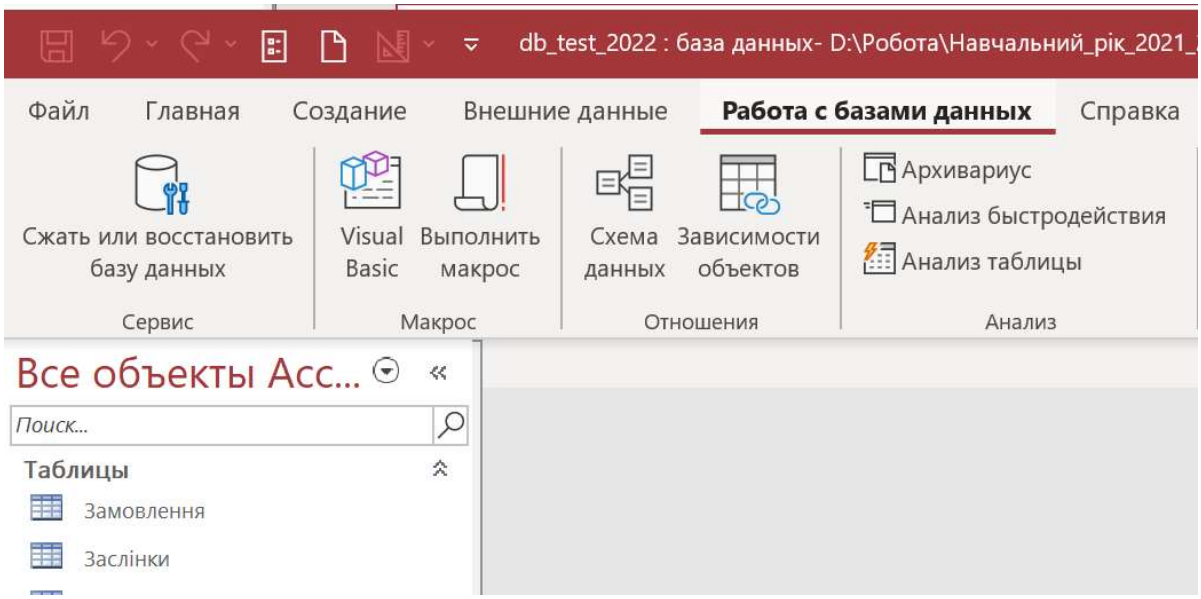



Рис. 1.13. Вікно режиму **Отношения** для створення **Схеми бази даних**

Контекстне меню вікна **Схема данных** має опції для створення та редагування зв’язків між таблицями бази даних.

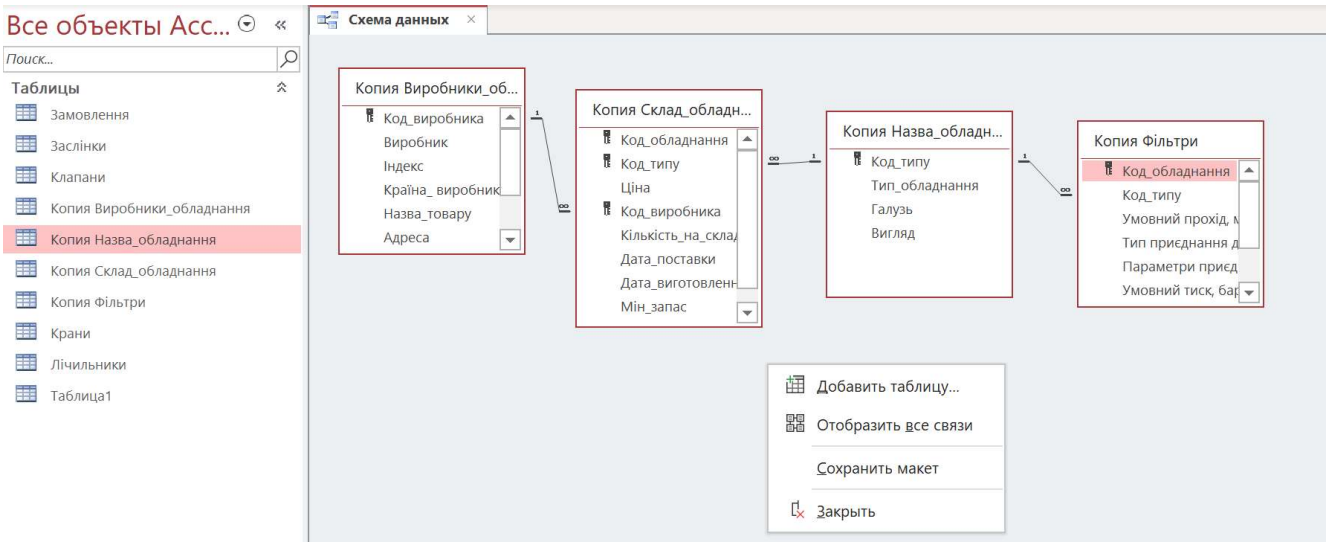



Рис. 1.14. Контекстне меню для створення та редагування зв’язків між таблицями бази даних

У вікні **Создание** необхідно обрати таблиці та поля за допомогою яких ці таблиці будуть поєднані. Необхідно зважити на те, що під час створення зв’язку між таблицями спільні поля можуть мати різні імена, хоча часто вони однакові. Але що дуже важливо, спільні поля мусять мати дані одного типу. Якщо поле первинного ключа має тип **Короткий текст**, то поле зовнішнього ключа також

мусить мати тип Короткий текст та обидва поля повинні мати однакові значення властивості Розмір поля.

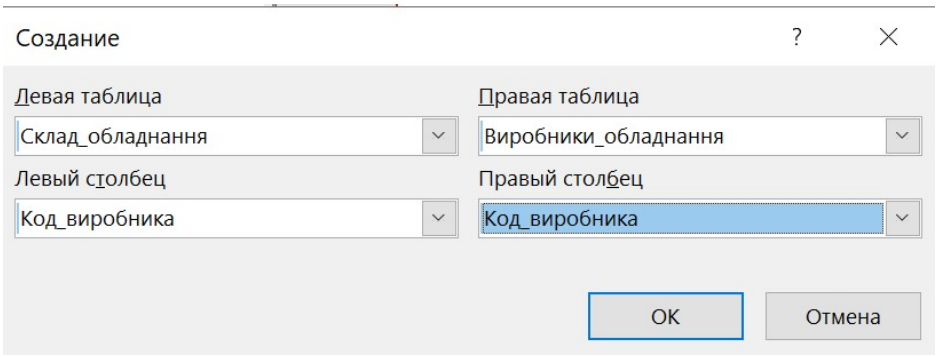



Рис. 1.15. Вікно вибору полів таблиць для поєднання таблиць

У вікні **Изменение связей** зазначено звідки і куди встановлено зв'язок, якщо типи даних полів, по яким виконується об’єднання таблиць збігаються (виконуються правила створення зв'язків), то в полі Тип отношения з'явиться повідомлення: один-до-багатьох (один-ко-многим).

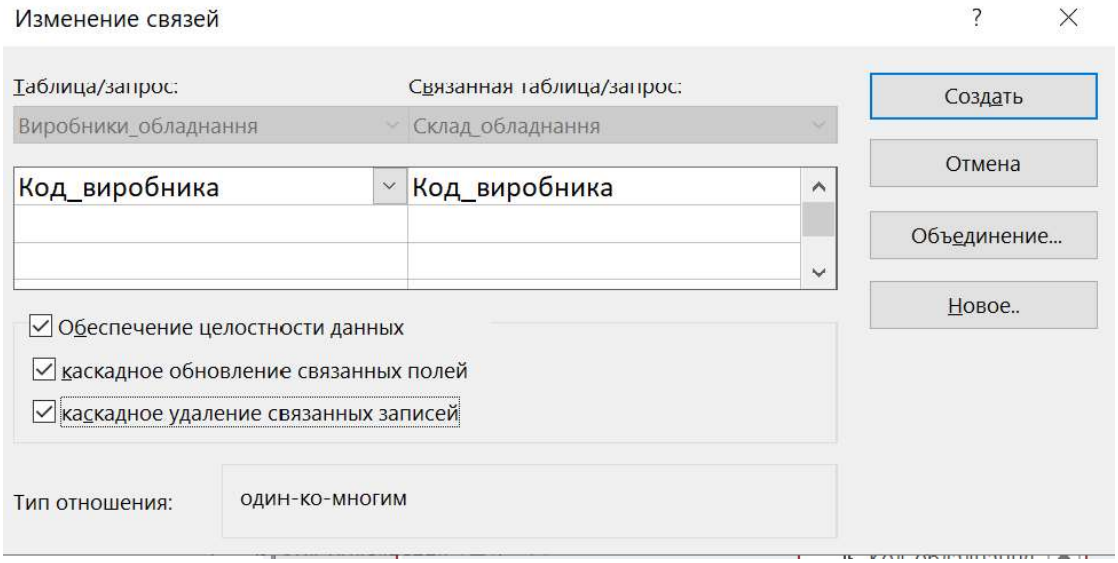



Рис.1.16. Вікно для встановлення параметрів зв'язків між таблицями

Після натискання кнопки **Создать** у вікні **Схема даних** з'явиться зв'язок. На одному кінці буде стояти 1 (одиниця) (ці дані є унікальними), а на іншому кінці зв'язку стоятиме знак − ∞ нескінченність).

Бажано зробити позначки і в інших полях, тоді при внесенні змін даних у вихідну таблицю при додаванні, редагуванні або їх видаленні, у всіх зв’язаних таблицях відбудеться автоматична зміна даних.

Для редагування чи видалення зв’язку необхідно виділити лінію, що позначає зв'язок та обрати з контекстного меню потрібну опцію **Изменить связь** чи **Удалить** (рис. 1.17).

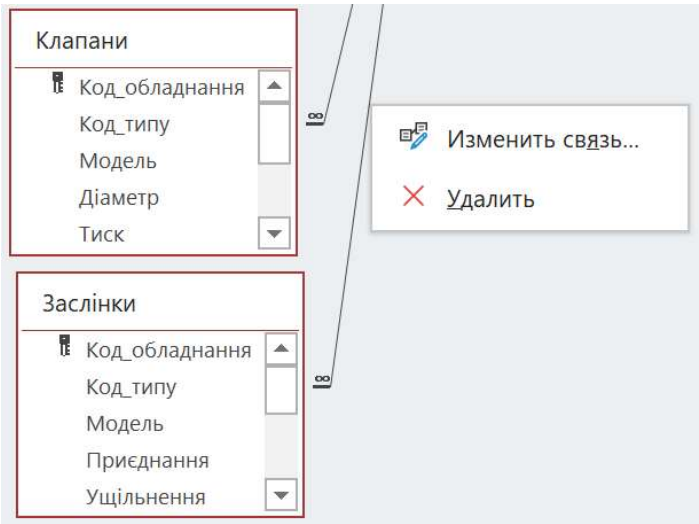



Рис. 1.17. Вікно з опціями контекстного меню для редагування зв’язків

## 1.4. Завдання до комп’ютерного практикуму

1. Створити нову папку з ім’ям FFF_ NNNN, де FFF – це прізвище студента. NNNN -шифр групи.
2. Створити нову базу даних та зберегти її з ім’ям FFF_DB_var_N_DDDD у папці FFF_ NNNN, де FFF- прізвище студента, var_N – номер варіанта індивідуального завдання, DDDD – рік виконання комп’ютерного практикуму.
3. Створити структуру таблиці варіанту індивідуального завдання. Надати таблиці ім’я Nazva_FFF, де Nazva-назва товару, FFF – це прізвище студента.
4. Додати поле «Код обладнання». Визначити це поле як ключове, за допомогою кнопки  .
5. Заповнити таблицю Nazva_FFF даними. Зберегти створену базу даних
6. Імпортувати для всіх варіантів з бази даних db_stud_comp_prak.accdd таблиці Виробники_обладнання, Замовлення, Назва_обладнання, Склад_обладнання, Покупець. Змінити імена цих таблиць, додавши через символ підкреслення , прізвище студента.

7. Імпортувати відповідно до свого варіанту з бази даних db_stud_comp_prak.accdd таблицю Лічильники (для варіантів 4,9,13,16), таблицю Клапани (для варіантів 1,8,11,17), таблицю Крани (для варіантів 6,10,14,18), Заслінки (для варіантів 3,12,15,19), таблицю Фільтри (для варіантів 2,5,7,20), використавши запит на створення таблиці. Зберегти імпортовані таблиці з іменами ЛічильникиN_FFF, КлапаниN_FFF, КраниN_FFF, ЗаслінкиN_FFF, ФільтриN_FFF. Де N – номер варіанта студента, FFF це прізвище студента!!!

8. Використати **Мастер подстановок** для введення даних текстового поля для таблиці індивідуального варіанта (варіанти таблиць наведено вище). Поле для введення даних з використанням **Мастер подстановок** обрати самостійно.

9. Створити в базі даних FFF_DB_var_N_DDDD таблицю для індивідуального завдання, що знаходяться нижче. Крім заданих полів, створити два нові поля Код_обладнання (або Код_товару) в залежності від вмісту таблиці та Код_типу.

10. Створити **Схему бази даних** відповідно до рис. 1.12. Додати зв'язок з таблицею індивідуального завдання.

11. Створену на цьому комп’ютерному практикумі базу даних потрібно використовувати на комп’ютерних практикумах 4-12. Виключенням є комп’ютерний практикум № 10.

12. Оформити звіт про виконання комп’ютерного практикуму, використовуючи редактор текстів WORD. Протокол повинен містити: назву, тему та мету комп’ютерного практикуму, варіант індивідуального завдання, а також опис створення застосування в СУБД Access (структури новоствореної таблиці індивідуального завдання, Схеми бази даних), повні письмові відповіді на контрольні запитання.

# 1.5. Індивідуальні завдання

Варіант 1

| № | Назва товару | Кількість | Ціна | Похибка вимірювання | Робочий тиск | Приєднання трубопроводу | Установка лічильників | Виробник | Країна виробника |
|---|---|---|---|---|---|---|---|---|---|
| 1 | BTPMGI-15 | 15 | 321,5 | 3 | 1,8 | Фланцеве | Сендвіч | Ukrdat | Ukraine |
| 2 | BTPMGI-20 | 20 | 412,5 | 2 | 1,6 | Різьбове | Горизонтально | Gydro | USA |
| 3 | BTPMGI-25 | 25 | 213,5 | 3 | 1,6 | Різьбове | Горизонтально | Radar | Poland |
| 4 | BTPMGI-32 | 32 | 361,0 | 3 | 1,6 | Різьбове | Горизонтально | Water | France |
| 5 | BTPMGI-40 | 40 | 110,6 | 1 | 2,5 | Фланцеве | вертикально | Gydro | USA |
| 6 | BTPMGI-50 | 50 | 415,7 | 2 | 2,5 | Фланцеве | вертикально | Ukrdat | Ukraine |

Варіант 2

| № | Назва товару | WAN-порт | Швидкість | Швид-кість | Ціна грн. | Кількість | Країна | Фірма |
|---|---|---|---|---|---|---|---|---|
| 1 | TL-WR841N | Ethernet | 300 | 100 | 529 | 7 | Китай | TP-LINK |
| 2 | TL-WR940N | Ethernet | >300 | 100 | 699 | 10 | Канада | TP-LINK |
| 3 | Archer C20 | Ethernet | >300 | 10/100 | 799 | 34 | Китай | TP-LINK |
| 4 | WF2780 | Ethernet | >300 | 1Гбіт/с | 1999 | 25 | Франція | Netis |
| 5 | WF2880 | Ethernet | >300 | 2Гбіт/с | 1055 | 12 | Франція | Netis |
| 6 | WF5 TP-Link | Ethernet | >300 | 2Гбіт/с | 899 | 25 | Китай | TP-LINK |

Варіант 3

| № | Товар | Закуплено, кг | Продано, кг | Ціна закупівлі грн.. | Ціна на Продаж грн.. | Фірма |
|---|---|---|---|---|---|---|
| 1 | Молібден | 3540 | 2558 | 360 | 382 | УкрІнвест |
| 2 | Мідь | 2452 | 1223 | 421 | 473 | ПромТех |
| 3 | Бронза | 5213 | 5005 | 522 | 555 | ПромТех |
| 4 | Чавун | 558 | 455 | 223 | 253 | МетІнвест |
| 5 | Срібло | 758 | 652 | 233 | 264 | Фаберже |
| 6 | Сталь | 4473 | 3354 | 175 | 190 | МетІнвест |
| 7 | Платина | 758 | 558 | 435 | 489 | Фаберже |

Варіант 4

| № | Товар | Марка (номінал) | Кількість | Ціна за 1 шт. | Рік випуску | Фірма |
|---|---|---|---|---|---|---|
| 1 | Транзистор | КТ-315 | 48 | 3,27 | 2020 | Sony |
| 2 | Транзистор | КТ-361 | 40 | 2,25 | 2019 | Samsung |
| 3 | Транзистор | КП-809 | 32 | 6,35 | 2020 | Hynix |
| 4 | Транзистор | КТ-816 | 29 | 4,35 | 2019 | Sony |
| 5 | Конденсатор | 5мкф | 48 | 6,23 | 2020 | Samsung |
| 6 | Конденсатор | 20мкф | 52 | 9,30 | 2022 | Hynix |
| 7 | Резистор | 1М | 59 | 10,11 | 2022 | Hynix |
| 8 | Резистор | 370К | 55 | 5,10 | 2019 | Sony |
| 9 | Резистор | 100К | 71 | 9,09 | 2021 | Samsung |
| 10 | Резистор | 20К | 25 | 8,08 | 2021 | Sony |

Варіант 5

| № | Товар | Закуплено, кг | Продано, кг | Ціна закупівлі | Ціна на продаж | Фірма |
|---|---|---|---|---|---|---|
| 1 | Залізо | 3540 | 2558 | 102 | 123 | МетІнвест |
| 2 | Мідь | 2452 | 1223 | 210 | 237 | УкрІнвест |
| 3 | Платина | 4452 | 4152 | 435 | 455 | Фаберже |
| 4 | Чавун | 2487 | 2187 | | | УкрІнвест |
| 5 | Бронза | 5213 | 5005 | 226 | 266 | УкрІнвест |
| 6 | Золото | 558 | 455 | 355 | 395 | Фаберже |
| 7 | Срібло | 758 | 672 | 333 | 346 | Фаберже |
| 8 | Сталь | 4473 | 3354 | 152 | 180 | МетІнвест |

Варіант 6

| Код товару | Назва товару | Кількість | Ціна | Похибка вимірювання | Робочий тиск | Приєднання трубопроводу | Установка лічильників | Виробник | Країна виробника |
|---|---|---|---|---|---|---|---|---|---|
| 1 | VERMGI-15 | 65 | 571,5 | 3 | 1,8 | Фланцеве | Сендвіч | Ukrdat | Ukraine |
| 2 | BTPMGI-20 | 50 | 452,5 | 2 | 1,6 | Різьбове | Горизонтально | Gydro | USA |
| 3 | GORMGI-25 | 45 | 223,5 | 3 | 1,6 | Різьбове | Горизонтально | Radar | Poland |
| 4 | BTPMGI-32 | 32 | 381,0 | 3 | 1,6 | Різьбове | Горизонтально | Water | Ukraine |
| 5 | BTPMGI-40 | 80 | 170,6 | 1 | 2,5 | Фланцеве | вертикально | Gydro | Poland |
| 6 | DTPMGI-50 | 50 | 495,7 | 2 | 2,5 | Фланцеве | вертикально | Ukrdat | USA |

Варіант 7

| № | Товар | Марка | Ціна | Фірма | Штук |
|---|---|---|---|---|---|
| 1 | Monitor | AOC C24G1/01 | 4751 | ASUS | 5 |
| 2 | | C27F396F | 3821 | Samsung | 4 |
| 3 | | BenQ EW2775ZH | 4932 | BenQ | 8 |
| 4 | | Acer Nitro QG221Qbii | 2799 | Aser | 0 |
| 5 | Відеокарта | MSI PCI-Ex GeForce | 1399 | Nvidia | 12 |
| 6 | | Ventus OC 8GB | 1347 | Nvidia | 14 |
| 7 | | Asus PCI-Ex | 2458 | ASUS | 9 |
| 8 | | Zotac PCI-Ex | 4489 | Samsung | 8 |
| 9 | | Asus PCI-Ultra | 2723 | ASUS | 3 |

Варіант 8

| № | Товар | Марка(номінал) | Кількість | Ціна за 1 шт. | Рік випуску | Фірма |
|---|---|---|---|---|---|---|
| 1 | Транзистор | КТ-315 | 48 | 5,27 | 2017 | Sony |
| 2 | Транзистор | КТ-361 | 40 | 6,25 | 2018 | Samsung |
| 3 | Транзистор | КП-809 | 32 | 2,35 | 2017 | Hynix |
| 4 | Транзистор | КТ-816 | 29 | 9,35 | 2018 | Sony |
| 5 | Конденсатор | 5мкф | 48 | 8,23 | 2017 | Samsung |
| 6 | Конденсатор | 20мкф | 52 | 3,30 | 2018 | Hynix |
| 7 | Резистор | 1М | 59 | 5,11 | 2019 | Hynix |
| 8 | Резистор | 370К | 55 | 7,10 | 2018 | Sony |
| 9 | Резистор | 100К | 71 | 7,09 | 2019 | Samsung |
| 10 | Резистор | 20К | 25 | 5,08 | 2019 | Sony |

Варіант 9

| Номер запису | Назва товару | Час-тота ГГц | Швидкість Wi-Fi, Мбіт/с | Швид-кість LAN, Мбіт/с | Ціна грн. | Кількість | Країна | Фірма |
|---|---|---|---|---|---|---|---|---|
| 1 | TL-WR841N | 2,4 | 300 | 100 | 829 | 37 | France | TP-LUK |
| 2 | TL-WR940N | 2,4 | 300 | 100 | 799 | 110 | Canada | TP-LINK |
| 3 | Archer C20 | 5 | 300 | 10/100 | 1099 | 214 | France | TP-LUK |
| 4 | WF2780 | 5 | 300 | 1Гбіт/с | 999 | 154 | Germany | Getis |
| 5 | WF2880 | 5 | 300 | 2Гбіт/с | 1455 | 96 | Germany | Getis |
| 6 | AC5400 | 5 | 500 | 2Гбіт/с | 2985 | 211 | Canada | TP-LINK |
| 7 | TP-Link Archer C9 | 5 | 500 | 2Гбіт/с | 2369 | 306 | France | D-LINK |

Варіант 10

| Код товару | Назва товару | Кількість | Ціна | Похибка вимірювання | Робочий тиск | Приєднання трубопроводу | Установка лічильників | Виробник | Країна виробника |
|---|---|---|---|---|---|---|---|---|---|
| 1 | BTPMGI-15 | 15 | 321,5 | 3 | 1,8 | Фланцеве | Сендвіч | Waterdat | Ukraine |
| 2 | BTPMGI-20 | 20 | 412,5 | 2 | 1,6 | Різьбове | Горизонтально | Didro | France |
| 3 | BTPMGI-25 | 25 | 213,5 | 3 | 1,6 | Різьбове | Горизонтально | Cadar | Poland |
| 4 | BTPMGI-32 | 32 | 361,0 | 3 | 1,6 | Різьбове | Горизонтально | Cadar | Poland |
| 5 | BTPMGI-40 | 40 | 110,6 | 1 | 2,5 | Фланцеве | вертикально | Didro | France |
| 6 | BTPMGI-50 | 50 | 415,7 | 2 | 2,5 | Фланцеве | вертикально | Waterdat | Ukraine |

Варіант 11

| Номер запису | Назва товару | Діаметр | Ціна | Кількість | Робочий тиск | Приєднання трубопроводу | Установка лічильників | Виробник | Країна виробника |
|---|---|---|---|---|---|---|---|---|---|
| 1 | BTPMGI-15 | 15 | 341,5 | 12 | 1,8 | Фланцеве | Сендвіч | Ukrdat | Ukraine |
| 2 | BTPMGI-20 | 20 | 452,5 | 23 | 1,6 | Різьбове | Горизонтально | Gydro | USA |
| 3 | BTPMGI-25 | 25 | 233,5 | 33 | 1,6 | Різьбове | Горизонтально | Radar | Poland |
| 4 | BTPMGI-32 | 32 | 456 | 13 | 1,6 | Різьбове | Горизонтально | Water | Ukraine |
| 5 | BTPMGI-40 | 40 | 510 | 18 | 2,5 | Фланцеве | вертикально | Gydro | USA |
| 6 | BTPMGI-50 | 50 | 415 | 25 | 2,5 | Фланцеве | вертикально | Ukrdat | Ukraine |

Варіант 12

| Номер запису | Назва товару | Час-тота ГГц | Швидкість Wi-Fi Мбіт/с | Швид-кість LAN Мбіт/с | Ціна грн. | Кількість | Країна | Фірма |
|---|---|---|---|---|---|---|---|---|
| 1 | TL-WR841N | 2,4 | 300 | 100 | 529 | 7 | Poland | TP-LINK |
| 2 | TL-WR940N | 2,4 | >300 | 100 | 699 | 10 | Canada | DataLink |
| 3 | Archer C20 | 2,4 і 5 | >300 | 10/100 | 799 | 34 | Poland | TP-LINK |
| 4 | WF2780 | 2,4 і 5 | >300 | 1Гбіт/с | 999 | 25 | France | Netis |
| 5 | WF2880 | 2,4 і 5 | >300 | 2Гбіт/с | 1055 | 12 | France | Netis |
| 6 | TL-WR940N | 5 | >300 | 2Гбіт/с | 2345 | 32 | Canada | DataLink |

Варіант 13

| Код товару | Назва товару | Кількість | Ціна | Похибка вимірювання | Робочий тиск | Приєднання трубопроводу | Установка лічильників | Виробник | Країна виробника |
|---|---|---|---|---|---|---|---|---|---|
| 1 | BTPMGI-15 | 45 | 321,5 | 3 | 1,8 | Фланцеве | Сендвіч | Ukrdat | Ukraine |
| 2 | BTPMGI-20 | 40 | 412,5 | 2 | 1,6 | Різьбове | Горизонтально | Gydro | USA |
| 3 | ATPMGI-25 | 55 | 253,5 | 3 | 1,6 | Різьбове | Горизонтально | Radar | Poland |
| 4 | BTPMGI-32 | 82 | 381,0 | 3 | 1,6 | Різьбове | Горизонтально | Water | France |
| 5 | KTPMGI-40 | 70 | 190,6 | 1 | 2,5 | Фланцеве | вертикально | Gydro | USA |
| 6 | BTPMGI-50 | 30 | 495,7 | 2 | 2,5 | Фланцеве | вертикально | Ukrdat | Ukraine |
| 7 | LKPMGI-30 | 60 | 455,7 | 1 | 1,8 | Різьбове | Сендвіч | Dedros | France |

Варіант 14

| Номер запису | Назва товару | Час-тота ГГц | Кількість портів | Швид-кість LAN Мбіт/с | Ціна грн. | Кіль-кість | Країна | Фірма |
|---|---|---|---|---|---|---|---|---|
| 1 | SOHO | 2,4 | 5 | 100/1000 | 429 | 7 | Ukraine | TENDA |
| 2 | TL-WR940N | 2,4 | 8 | 100 | 799 | 10 | Canada | TP-LINK |
| 3 | D-Link DES-1008C | 5 | 16 | 10/100 | 1099 | 24 | France | Desktop |
| 4 | D-Link DGS-1024A | 5 | 24 | 1Гбіт/с | 999 | 15 | France | D-LINK |
| 5 | WF2880 | 5 | 16 | 2Гбіт/с | 1455 | 9 | Germany | Getis |
| 6 | AC5400 | 5 | 24 | 2Гбіт/с | 2985 | 21 | Canada | TP-LINK |
| 7 | D-Link DES-1008C | 5 | 48 | 2Гбіт/с | 2369 | 30 | France | D-LINK |

Варіант 15

| Номер запису | Назва товару | Час-тота ГГц | Швидкість Wi-Fi, Мбіт/с | Швид-кість LAN Мбіт/с | Ціна грн. | Кіль-кість | Країна | Фірма |
|---|---|---|---|---|---|---|---|---|
| 1 | TL-WR841N | 2,4 | 300 | 100 | 1829 | 7 | France | TP-LUK |
| 2 | TL-WR940N | 2,4 | 300 | 100 | 1099 | 10 | Canada | TP-LINK |
| 3 | Archer C20 | 5 | 300 | 10/100 | 1099 | 24 | France | TP-LUK |
| 4 | ASUS780 | 5 | 300 | 1Гбіт/с | 1379 | 15 | Ukraine | ASUS |
| 5 | ASUS880 | 5 | 300 | 2Гбіт/с | 2455 | 11 | Ukraine | ASUS |
| 6 | AC5400 | 5 | 500 | 2Гбіт/с | 2985 | 21 | Canada | TP-LINK |
| 7 | TP-Link Archer C9 | 5 | 500 | 2Гбіт/с | 3369 | 30 | Canada | D-LINK |

Варіант 16

| Номер запису | Назва товару | Час-тота (ГГц) | Швидкість Wi-Fi (Мбіт/с) | Швидгість LAN (Мбіт/с) | Ціна (грн) | Кількість | Країна | Фірма |
|---|---|---|---|---|---|---|---|---|
| 1 | TP-LINK SG105 | 2,4 | 300 | 100 | 934 | 7 | Poland | D-LINK |
| 2 | TL-WR940N | 2,4 | >300 | 100 | 899 | 10 | Canada | DataLink |
| 3 | Mercusys MS108 | 2,4 і 5 | >300 | 10/100 | 1199 | 34 | Poland | TP-LINK NetWork |
| 4 | Tenda S105 | 2,4 і 5 | >300 | 1Гбіт/с | 138 | 25 | France | ASUS |
| 5 | TP-LINK TL-SF1024 | 2,4 і 5 | >300 | 2Гбіт/с | 1055 | 12 | France | Netis |
| 6 | TL-WR940N | 5 | >300 | 2Гбіт/с | 2345 | 32 | Canada | DataLink |

Варіант 17

| Назва товару | Кількість | Ціна | Похибка вимірювання | Робочий тиск | Приєднання трубопроводу | Установка лічильників | Виробник | Країна виробника |
|---|---|---|---|---|---|---|---|---|
| VERMGI-15 | 65 | 571,5 | 3 | 1,8 | Фланцеве | Сендвіч | Ukrdat | Ukraine |
| BTPMGI-20 | 50 | 452,5 | 2 | 1,6 | Різьбове | Горизонтально | Gydro | USA |
| GORMGI-25 | 45 | 223,5 | 3 | 1,6 | Різьбове | Горизонтально | Radar | Poland |
| BTPMGI-32 | 32 | 381,0 | 3 | 1,6 | Різьбове | Сендвіч о | Water | Ukraine |
| BTPMGI-40 | 80 | 170,6 | 1 | 2,5 | Фланцеве | вертикально | Gydro | Poland |
| DTPMGI-50 | 50 | 495,7 | 2 | 2,5 | Фланцеве | вертикально | Ukrdat | Ukraine |
| LTPMGI-36 | 80 | 495,7 | 1 | 1,6 | Різьбове | вертикально | Dadro | France |
| ZTPMGI-38 | 80 | 495,7 | 1 | 1,8 | Різьбове | Сендвіч | Gidro | USA |

Варіант 18

| Назва товару | Кількість портів | Частота ГГц | Швидкість Wi-Fi, Мбіт/с | Швид-кість LAN, Мбіт/с | Ціна грн. | Кіль-кість | Країна | Фірма |
|---|---|---|---|---|---|---|---|---|
| TL-WR841N | 8 | 2,4 | 300 | 100 | 1829 | 7 | France | TP-LUK |
| TL-WR940N | 12 | 2,4 | 300 | 100 | 1099 | 10 | Canada | TP-LINK |
| Archer C20 | 8 | 5 | 300 | 10/100 | 1099 | 24 | France | TP-LUK |
| ASUS780 | 24 | 5 | 300 | 1Гбіт/с | 1379 | 15 | Ukraine | ASUS |
| ASUS880 | 24 | 5 | 300 | 2Гбіт/с | 2455 | 11 | Ukraine | ASUS |
| AC5400 | 8 | 5 | 500 | 2Гбіт/с | 2985 | 21 | Canada | TP-LINK |
| TP-Link Archer C9 | 12 | 5 | 500 | 2Гбіт/с | 3369 | 30 | Canada | D-LINK |
| TL-SG1008D | 8 | 2,4 | 300 | 1Гбіт/с | 555 | 60 | USA | TP-LINK NetWork |
| Archer TL-20 | 48 | 2,4 | 300 | 1Гбіт/с | 5786 | 40 | France | TP-LUK |

## 1.6. Контрольні запитання

1. Для чого використовується об’єкт бази даних таблиця?
2. Які способи створення таблиць існують в СУБД Access?
3. Які способи заповнення таблиць даними існують в СУБД Access?
4. Які типи даних існують в СУБД Access?
5. Які формати даних можуть використовуватися типом даних «Числовой»?
6. Які формати даних можуть використовуватися типом даних «Дата/время»?
7. Які формати даних можуть використовуватися типом даних «Текстовый»?
8. Які формати даних можуть використовуватися типом даних «Денежный»?
9. Які формати даних можуть використовуватися типом даних «Логический»?
10. Які формати даних можуть використовуватися типом даних «Поле объекта OLE»?
11. Які формати даних можуть використовуватися типом даних «Гиперссылка»?
12. Детально опишіть властивості полів таблиці для кожного типу (призначення, можливі формати і т.і.), наведіть приклади.
13. Опишіть призначення ключового поля.
14. Чим відрізняється операція імпортування від операції копіювання таблиць в СУБД Access? В чому полягає ця відмінність?
15. Для чого призначена схема даних в СУБД Access?
16. Які типи зв’язків між таблицями існують в СУБД Access?

# Комп'ютерний практикум № 2. Використання мови SQL для роботи з базою даних

*Мета роботи* – навчитися використовувати команди мови SQL для роботи з базами даних.

## Теоретичні відомості

Мова структурованих запитів - Structured Query Language (SQL) стала найпоширенішою та фактично перетворилася на стандартну мову реляційних баз даних.

Стандарт мови SQL було розроблено Американским національним інститутом стандартів (ANSI) в 1986 році, а в 1987 році Міжнародна організація стандартів (ISO) прийняла цей стандартів у якості міжнародного.

Мова SQL використовується десятками СУБД різноманітних типів, розроблених для різних обчислювальних платформ, починаючи від персональних комп’ютерів і закінчуючи мейнфреймами.

Мова SQL надає користувачу наступні можливості:

- Створення бази даних і таблиць з повним описом структури таблиць.
- Виконання основних операцій маніпулювання даними (вставка, видалення, модифікація даних з таблиць).
- Виконання простих і складних запитів для отримання інформації з бази даних.

Мова SQL складається з наступних основних компонентів:

SQL DDL (Data Definition Language ) – призначеної для визначення структури бази даних і керування доступом до неї;

SQL DML (Data Manipulation Language) – призначеної для виконання дій над даними;

SQL DQL (Data Query Language) – призначеної для виконання дій для вибору даних з таблиць.

Мова SQL DDL дозволяє створювати та знищувати різні об'єкти бази даних - наприклад, схеми, домени, таблиці, індекси.

Команди SQL DDL

CREATE TABLE – за допомогою цієї команди можна створювати структуру таблиці з описом всіх атрибутів та їх властивостей;

ALTER TABLE – за допомогою цієї команди можна модифікувати структуру таблиці;

DROP TABLE – за допомогою цієї команди можна видаляти таблицю;

CREATE INDEX – за допомогою цієї команди можна створювати індекс;

ALTER INDEX – за допомогою цієї команди можна модифікувати індекс;

DROP INDEX – за допомогою цієї команди можна видаляти індекс.

SQL DML (Data Manipulation Language)

SQL DML –призначена для виконання дій над даними (додавання, оновлення, видалення даних та ін.)

INSERT — вставка даних в таблицю;

UPDATE — оновлення (зміна) даних в таблиці;

DELETE — видалення даних з таблиці.

SQL DQL (Data Query Language)

SQL DQL – призначена для виконання дій для вибору даних з таблиць.

За допомогою команди SELECT мови SQL DQL в СУБД створюються запити.

Запит – це звернення до бази даних за відповідною інформацією.

Формат команди SELECT

Загальний формат команди SELECT має наступний вигляд:

SELECT [DISTINCT | ALL] {* | [col_exp [AS new_name] ] [,...]}

FROM table_name [alias], [ . . . ]

[WHERE condition]

[GROUP BYcol_list] [HAVING condition]

[ORDER BY col_list]

Параметр col_exp є ім'ям стовпця або виразом з декількох імен.

Параметр table_name є ім'ям існуючих в базі даних таблиці або запиту, до даних яких необхідно отримати доступ.

Необов'язковий параметр alias — це скорочення, яке можна встановити для імені таблиці table_name.

Послідовність виконання параметрів команди SELECT

FROM - Визначаються імена таблиці або декількох таблиць, що будуть використовуватися;

WHERE - Виконується фільтрація рядків об'єкту відповідно до заданих умов;

GROUP - Утворюються групи рядків, що мають одне і те ж значення у вказаному стовпчику;

HAVING - Фільтруються групи рядків об'єкту відповідно до вказаної умови.

SELECT-Встановлюється, які стовпці повинні бути присутніми у вихідних даних;

ORDER - Визначається впорядкованість результатів виконання команди.

В загальному випадку рядки в результуючій таблиці SQL-запиту не впорядковані яким-небудь певним чином. Проте їх можна необхідним чином відсортувати, для чого в оператор SELECT поміщається параметр ORDER. Параметр ORDER включає список розділених комами ідентифікаторів стовпців, по яких вимагається упорядкувати результуючу таблицю запиту. Ідентифікатором стовпця може бути або його ім'я, або номерN, який ідентифікує елемент списку SELECT та його позицію в цьому списку. Найлівіший елемент списку має номер 1, наступний — номер 2 і т.д. Номери стовпців можуть використовуватися в тих випадках, коли стовпці, по яких слід упорядкувати результат, є обчислюваними, а параметр AS з вказівкою імені цього стовпця в операторі SELECT відсутня. Параметр ORDER дозволяє упорядкувати вибрані записи в порядку зростання (ASC) або убування (DESC) значень будь-якого стовпця або комбінації стовпців, незалежно від того, присутні ці стовпці в таблиці результатів чи ні. Проте в деяких діалектах вимагається, щоб параметр ORDER обов'язково була присутня в списку вибірки оператора SELECT.

Підсумок по даним кожного стопця таблиці розташовується в одному узагальнюючому рядку. Проте дуже часто в звітах вимагається формувати і проміжні підсумки. Для цієї мети в операторі SELECT може використовуватися параметр GROUP. Запит, в якому присутня параметр GROUP, називається групуючим запитом, оскільки в ньому групуються дані, отримані в результаті виконання оператора SELECT, після чого для кожної окремої групи створюється

єдиний сумарний рядок. Стовпці, перераховані у фразі GROUP, називаються груповаными стовпцями. Стандарт ISO вимагає, щоб оператор SELECT і параметр GROUP були тісно зв'язані між собою. При використанні в операторі SELECT фрази GROUP кожен елемент списку в операторі SELECT повинен мати єдине значення для всієї групи. Більш того, оператор SELECT може включати тільки наступні типи елементів:

• імена стовпців;

• узагальнюючі функції;

• константи;

• вирази, які включають комбінації з перерахованих вище елементів.

Всі імена стовпців, наведені в списку оператора SELECT, повинні бути присутні і у фразі GROUP — за винятком випадків, коли ім'я стовпця використовується в узагальнюючій функції. Зворотне правило не є справедливим — у фразі GROUP можуть бути присутні імена стовпців, відсутні в списку оператора SELECT. Якщо спільно з фразою GROUP використовується параметр WHERE, то воно обробляється першим, а групуванню піддаються тільки ті рядки, які задовольняють умові пошуку.

Стандартом ISO визначено, що при проведенні групування всі відсутні значення розглядаються як рівні. Якщо два рядки таблиці в одному і тому ж групованому стовпці містять значення NULL і ідентичні значення у всій решті непорожніх групованих стовпців, вони поміщаються в одну і ту ж групу.

Параметр Having призначений для використовування спільно з фразою GROUP для завдання обмежень, що вказуються з метою відбору тих груп, які будуть розміщені в результуючій таблиці запиту. Хоча параметр HAVING і параметр WHERE мають схожий синтаксис, за призначенням вони відрізняються. Параметр WHERE призначений для фільтрації окремих рядків, що використовуються для групування або для розміщення в результуючій таблиці запиту, тоді як параметр HAVING використовується для фільтрації груп, що розміщуються в результуючій таблиці запиту. Стандарт ISO вимагає, щоб імена стовпців, що використовуються у фразі HAVING, обов'язково були присутні в списку фрази GROUP або застосовувалися в узагальнюючих функціях. На практиці умови пошуку у фразі HAVING завжди включають,

щонайменше, одну узагальнюючу функцію, в протилежному випадку ці умови пошуку повинні бути розташовані у параметрі WHERE і застосовуватися для відбору окремих рядків. (Не забувайте, що узагальнюючі функції не можуть використовуватися в фразі WHERE.)

## 2.1. Оператори для запису логічних і арифметичних виразів

В мові SQL використовуються наступні групи операторів для запису логічних і арифметичних виразів:

- Оператори порівняння (=,<>,>,<,>=,<=)
- Логічні оператори (IS NULL, BETWEEN, IN, LIKE і т.і.)
- Логічні зв'язки (AND, OR, NOT).
- Оператори заперечення (IS NOT NULL, NOT BETWEEN , NOT IN, NOT LIKE, NOT EXIST , NOT UNIQUE).
- Арифметичні оператори +, - ,* ,/.

**Приклад 1.** Застосовуючи мову запитів SQL до відношення(таблиці) СК9 організуйте вибір інформації про комплектуючі фірми Danfoss , які зберігаються на складі і поставки яких припинено. Таблиця результатів повинна складатися з полів Тип_обладнання, Виробник, Поставки_припинено.

```
SELECT СК9.Тип_обладнання, СК9.Виробник, СК9.Поставки_припинено
    FROM СК9
    WHERE (СК9.Виробник=" Danfoss") AND
(СК9.Поставки_припинено=True);
```

Для того, що отримати результат з використанням наведеної вище команди, необхідно цю команду вставити в код програми, реалізованій з використанням алгоритмічної мови високого рівня, наприклад, мови C#. А потім запустити цей код на виконання. Код програми мовою C# має вигляд:

```
 private void button1_Click(object sender, EventArgs e)
 {
 //створимо запит, який повертатиме значення трьох стовпців для всіх
рядків, що
 // відповідають заданій умові
string query = " SELECT Тип_обладнання, Виробник, Поставки_припинено
```

```
FROM СК9
WHERE (Виробник=" Danfoss") AND (Поставки_припинено=True)";
OleDbCommand command = new OleDbCommand(query, myConnection);
OleDbDataReader reader = command.ExecuteReader();
listBox1.Items.Clear();
while (reader.Read())
{
// задається шаблон виведення даних, тобто до кожного рядка буде
виводитися три значення
listBox1.Items.Add(reader[0].ToString() + " " + reader[1].ToString() + " " +
reader[2].ToString() + " ");
}
reader.Close();
}
```

Запускатися на виконання цей код буде подією натиснення кнопки button1 відповідно попередньо створеної користувачем форми. Результат буде виводитися з використанням об'єкту listBox1.

## 2.2. Формат команди UPDATE

Команда UPDATE дозволяє змінювати вміст уже існуючих рядків зазначеної таблиці.

Загальний формат команди UPDATE має наступний вигляд:

***UPDATE table_name SET column_name1 = data_value1 [, column_name2 = data_value2 ]***

***[WHERE search_condition ]***

***Параметр WHERE є необов'язковим.***

- Якщо він опущений, значення зазначених стовпців будуть змінені у всіх рядках таблиці.
- Якщо параметр WHERE є присутнім, то оновленими будуть тільки ті рядки, які задовольняють умові пошуку, заданій у параметрі search_condition

**Приклад 2**. Всьому персоналу підвищити заробітну плату на 3%.

```
UPDATE Персонал
SET Заробітня_плата = Заробітня_плата*1.03;
```

## 2.3. Формат команди DELETE

Команда DELETE дозволяє видаляти рядки даних із зазначеної таблиці. Ця команда має наступний формат:

***DELETE FROM table_name [WHERE search_condition]***

Параметри команди DELETE:

- table_name містить ім'я таблиці бази даних;
- search_condition є необов'язковим параметром. За умови, якщо він опущений, з таблиці будуть вилучені всі існуючі в ній рядки.
- Якщо присутній параметр WHERE , то з таблиці будуть вилучені тільки ті рядки, які задовольняють умові відбору, заданій у search condition.

Зверніть увагу, що сама таблиця видалена не буде. Залишиться структура таблиці.

**Приклад 3**. Застосовуючи мову запитів SQL до відношення СУ організуйте видалення даних про комплектуючі, мінімальний запас яких на складі дорівнює 18 або 20.

***DELETE***
***FROM СУ***
***WHERE (Мін_запас=18 Or Мін_запас=20);***

## 2.4. Формат команди INSERT

Для додавання нових даних у таблиці існує два формата команди INSERT. Перша призначена для вставки єдиного рядка в зазначену таблицю.

Друга форма команди INSERT дозволяє скопіювати множину рядків однієї таблиці в іншу таблицю.

Команда INSERT першої форми має наступний формат:

***INSERT INTO table name [(column_list)] VALUES (data value list)***

Оператор INSERT другої форми має наступний формат:

***INSERT INTO table_name [(column_list)] SELECT . . .***

Параметри оператора INSERT.

table name це ім'я таблиці бази даних;

column_list це список, що складається з імен одного або більше стовпців, розділених комами; column_list є необов'язковим.

data value list - повинен бути записаний у такий спосіб, щоб відповідати параметру column list.

А саме :

- кількість елементів в обох списках повинна бути однаковою;
- повинна існувати пряма відповідність між позицією одного й того елемента в обох списках (перший – першому);
- типи даних елементів списку data_value list повинні бути сумісними з типом даних відповідних стовпців таблиці.

**Приклад 4.** Використання конструкції INSERT…VALUES

Занести в таблицю Персонал новий запис, що містить дані у всіх стовпцях

***INSERT INTO Персонал***

***VALUES (“SG16”, “Катерина”, “Стратієнко”, “Київ ”, “ Янгеля 25, кв. 34”,“044-211-3001”, “Асистент”, “Ж”, 8300,“B3”);***

*Зауваження*

Якщо розміщувати дані, якими мають бути заповнені стовпці таблиці, у порядку їхнього розташування в таблиці, то вказувати список аргументів стовпців необов'язково.

Значення типу Текстовий (наприклад, “Київ ”) повинні бути взяті в лапки.

**Приклад 5.** Ввести в таблицю Персонал новий запис, що містить дані про всі обов’язкові стовпці: Код, Ім’я, Прізвище, Посада, Заробіня_плата, Відділення.

Вигляд конструкції INSERT…VALUES

***INSERT INTO Персонал (Код, Ім’я, Прізвище, Посада, Заробітня_плата, Відділення) VALUES (“SG44”, “Катерина”, “Стратієнко”, “Асистент”, 8300, “B3”);***

Оскільки дані вставляються тільки в деякі стовпці таблиці, необхідно вказати імена стовпців, у яких ці дані будуть розташовуватися. Порядок розташування імен стовпців не є суттєвим, однак більш природньо вказувати їх у тому порядку, у якому вони знаходяться у таблиці.

Оператор INSERT можна було б записати й у такий спосіб:

***INSERT INTO Персонал***

***VALUES (“SG44”, “Катерина”,***

***“Стратієнко”, NULL, NULL, “Асистент”, NULL, NULL, 8300, “B3”);***

У цьому випадку явно вказано, що в стовпці Адреса, Номер_телефону, Стать повинні бути записані значення

NULL

**Приклад 6.** Записати до стовпців таблиці ЗвітТабл : Код_обладнання,

Код_типу, Тип, [Умовний прохід, мм], Ціна, Кількість, Виробник дані з відповідних стовпців віртуальної таблиці Сортувати_за_виробником_запит_кран, що відповідають введенному значенню – назві фірми виробника.

***INSERT INTO ЗвітТабл ( Код_обладнання, Код_типу,***

***Тип, [Умовний прохід, мм], Ціна, Кількість, Виробник )***

***SELECT Сорт.Код_обладнання, Сорт.Код_типу,***

***Сорт.Тип, Сорт.[Умовний прохід, мм], Сорт.Ціна,***

***Сорт.Кількість, Сорт.Виробник***

***FROM Сортувати_за_виробником_запит_кран Сорт;***

В цьому прикладі замість довгого імені таблиці Сортувати_за_виробником_запит_кран використовується аліас Сорт.

## 2.5. Використання команд мови SQL

Використання команд мови SQL необхідно реалізувати для створених таблиць бази даних Access.

Для зручної роботи з аналізу даних необхідно створити інтерфейс у середовищі Visual Studio C#.

Інтерфейсом у середовищі Visual Studio C# слугує форма Windows Forms, яка дозволяє обрати для проекту такі компоненти, як діалогові вікна, меню, кнопки і багато інших елементів керування, що є частиною стандартного користувальницького інтерфейсу (UI) Windows. По суті, ці елементи управління є просто класами з бібліотеки NET Framework. В режимі Конструктор в Visual C # існує можливість перетягувати елементи керування на основну форму програми та змінювати їх розміри і розташування. Після цього IDE автоматично

додають вихідний код для створення та ініціалізації екземпляра відповідного класу.

## 2.6. Створення проєкту у Microsoft Visual Studio 2017C#

1. У меню Файл виберіть команду Створити проект (New Project).

Відкриється діалогове вікно Створення проекту. У цьому діалоговому вікні виводиться список різних типів додатків за замовчуванням, які можна створювати за допомогою Visual Studio C # .

2. Виберіть Windows Forms Application в якості типу проекту.

3. Змініть ім'я проекту на DB_Test_SQL_FFF_var_N_DDDD, де FFF-прізвище студента, var_N – номер варіанта індивідуального завдання, DDDD – рік виконання комп'ютерного практикуму та збережіть його у папці FFF_NN ( створена на попередньому практикумі).

4. Натисніть кнопку ОК.

Створення проєкту у Microsoft Visual Studio 2019 C#

Після завантаження Visual Studio 2019 створюємо новий проєкт за допомогою команди Create new project

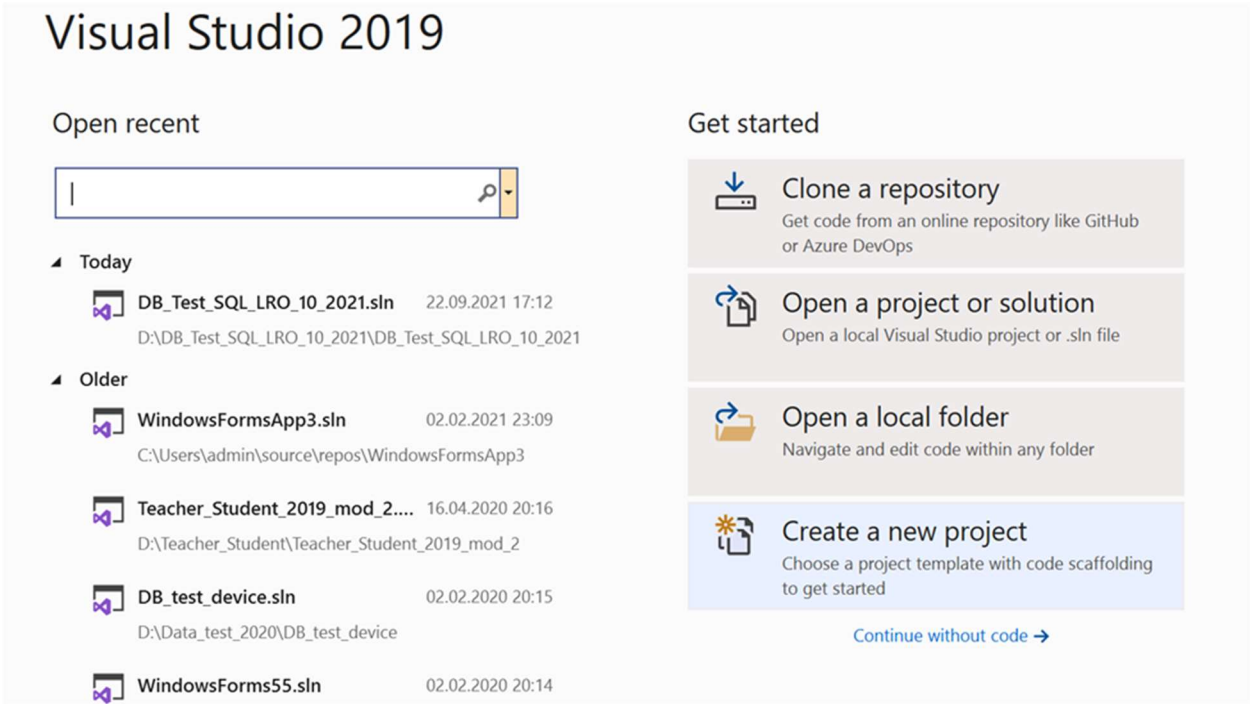



Рис. 2.1. Вікно Visual Studio 2019 для вибору роботи над проєктом

Після обрання команди Create new project, відкривається вікно з можливістю вибору вигляду новостворюваного застосування. Необхідно обрати Windows Forms App(.Net Framework).

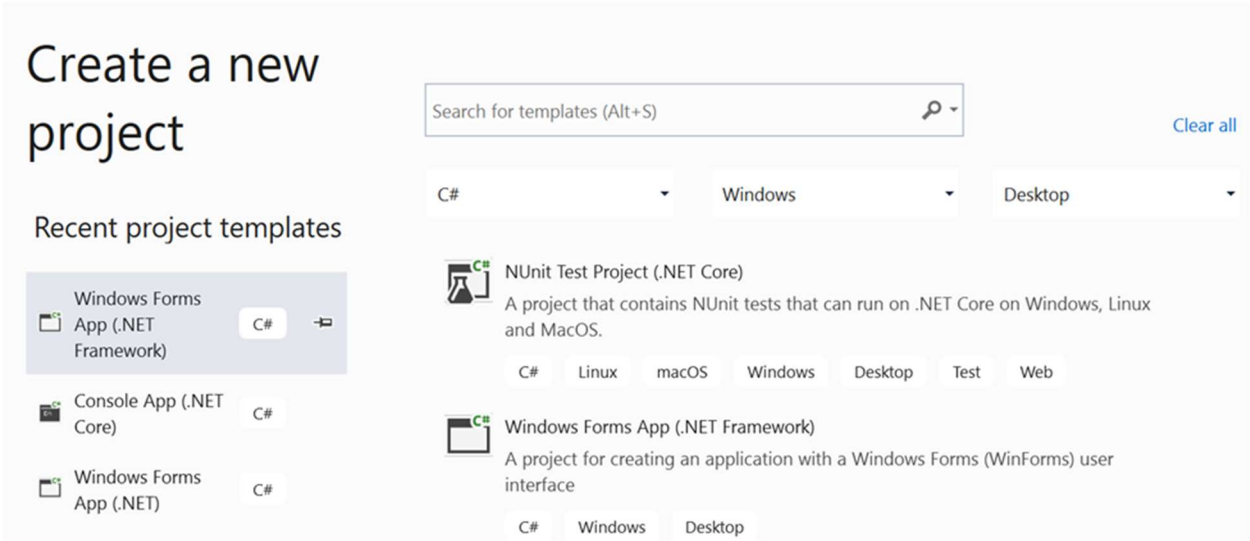



Рис. 2.2. Вікно Visual Studio 2019 для вибору вигляду застосування

Після чого відкривається вікно, в якому потрібно задати ім’я проєкту та визначити папку для зберігання всіх файлів проєкту.

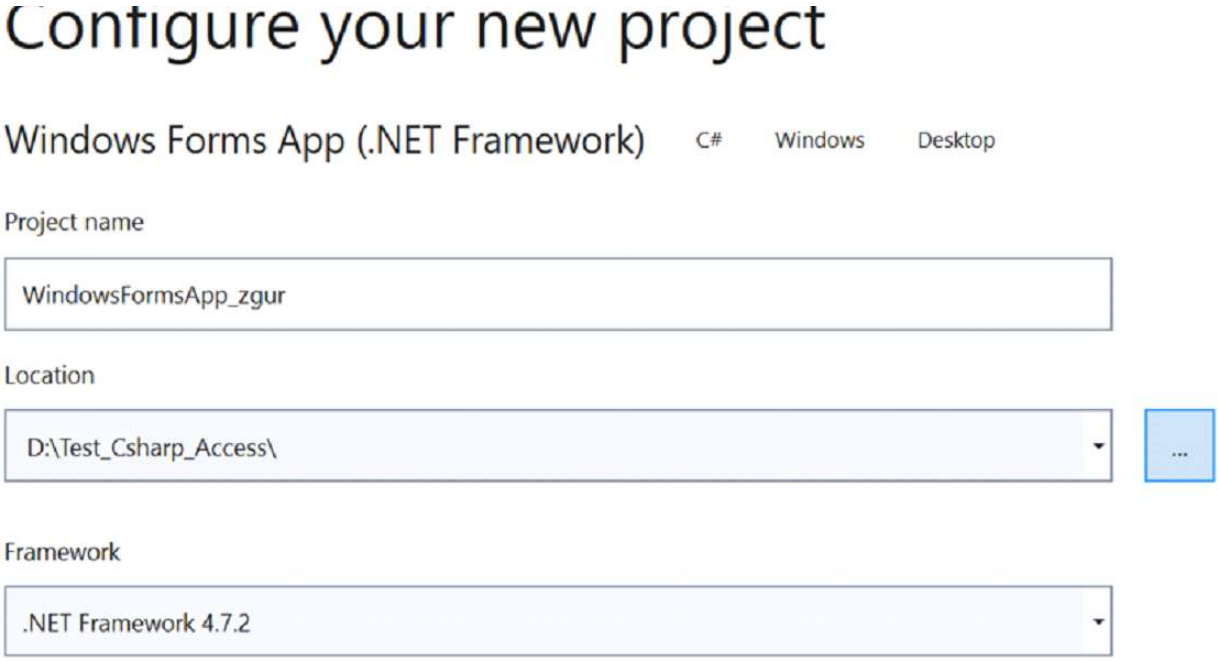



Рис. 2.3. Вікно Visual Studio 2019 для вибору папки зберігання файлів проєкту

Новий проєкт створено

## 2.7. Створення форми проєкту

В обох варіантах Visual C # створить для проєкту нову папку з таким же ім'ям, як у проєкту, і потім відобразить нову форму Windows з ім'ям From1 в режимі Конструктор. . Завантажена порожня форма і код, в якому описано властивості класу Form. Тепер необхідно модифікувати форму у відповідності до завдання комп'ютерного практикуму. Для роботи з формою використовуються два режими: режим Дизайн та режим Код. Перемикатися між режимом Дизайн та режимом Код можна клацнувши правою кнопкою миші поверхню розробки форми і вибравши команду Проглянути код (View Code) або Відкрити в конструкторі (View Designer) (рис. 2.4).

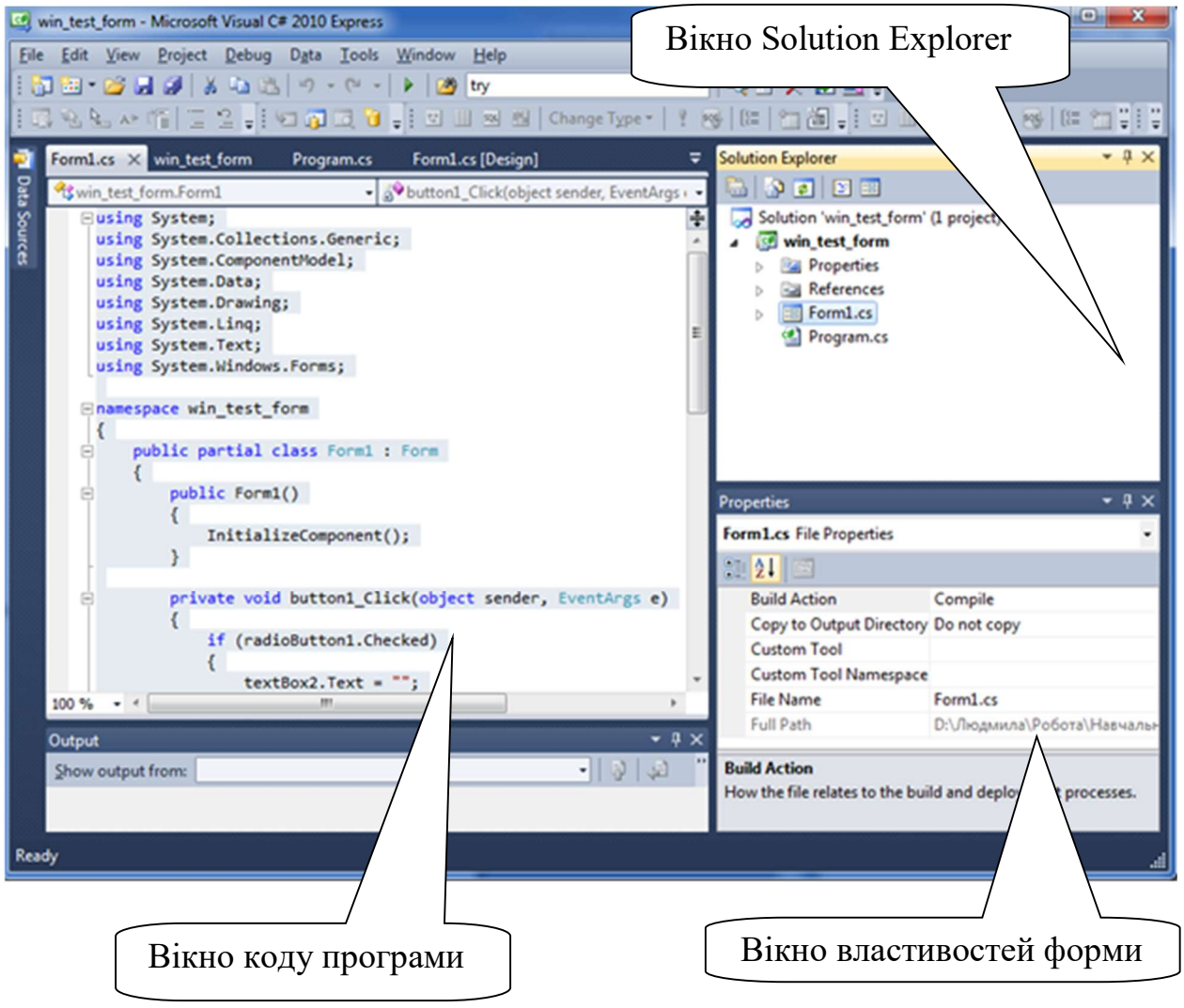



Рис. 2.4. Вікно Microsoft Visual Studio в режимі перегляду коду програми

Форма Windows в режимі Конструктор - це візуальне подання вікна, яке з'явиться при відкритті програми. У режимі Конструктор можна перетягувати

різні елементи управління з панелі елементів ToolBox у форму. Ці елементи управління не є реальними об'єктами, це просто зображення, які зручно переміщувати у формі для певного розташування.

Після розташування елемента керування у формі Visual C # у фоновому режимі створить код для правильного розміщення реального елемента керування при виконанні програми. Цей вихідний код буде знаходитися у файлі, який зазвичай не видно в застосуванні. Файл має ім'я Form1.designer.cs знаходиться в оглядачі рішень (Solution Explorer), якщо розгорнути вузол Form1.cs.

На наступному рисунку (рис. 2.5) представлено контекстне меню з командами Проглянути код (View Code) та Відкрити в конструкторі (View Designer) (рис. 2.6).

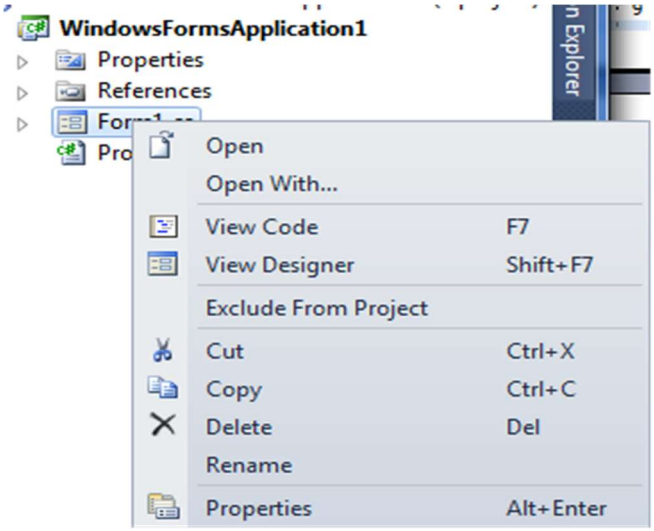



Рис. 2.5 . Вікно з контекстним меню при розгортанні вузла Form1.cs

У вікні Toolbox (його можна відобразити на екрані за допомогою команди меню View/Toolbox) знаходиться список елементів управління, які можна використовувати на формах програми. Тобто, який набір компонентів доступний в даний момент, залежить від типу проекту, що розробляється. Наприклад, якщо в даний момент розробляється програма типу Windows Forms, в цьому вікні будуть присутні елементи керування, які можна використовувати в Windows-застосуваннях; якщо ж розробляється Web-форма, в цьому вікні будуть знаходитися інструменти для роботи з елементами управління Web Controls, і т.д.

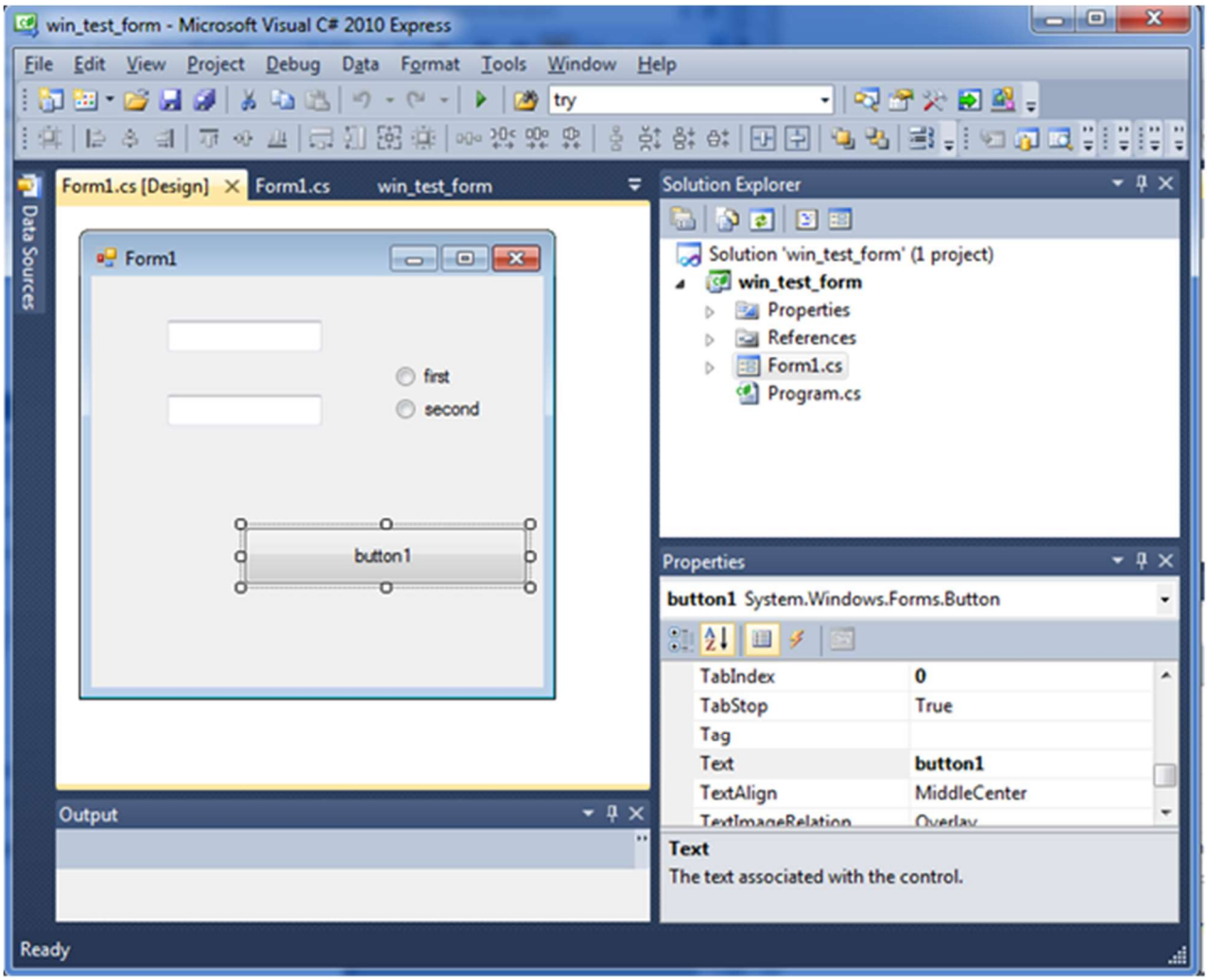



Рис. 2.6. Вікно Microsoft Visual Studio в режимі конструктора форми

При необхідності можна змінити відображений у вікні Toolbox набір елементів управління, додавши інші компоненти. NET або елементи ActiveX (у тому числі створені незалежними виробниками). Для цієї мети можна використовувати команду меню Tools/Choose Toolbox Items і за допомогою діалогової панелі Customize Toolbox (рис. 2.7) вибрати елементи керування ActiveX або елементи керування. NET, які ми хочемо відобразити у вікні Toolbox.

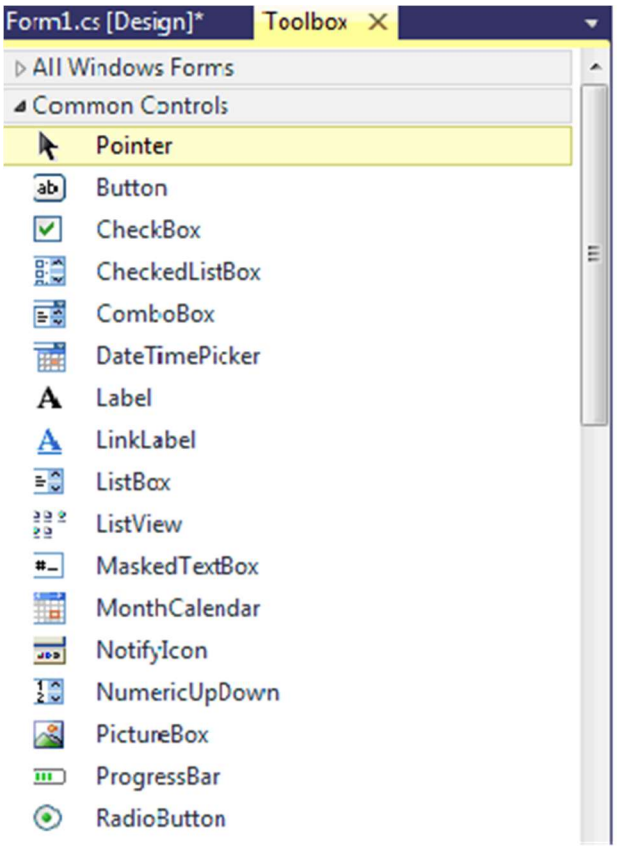
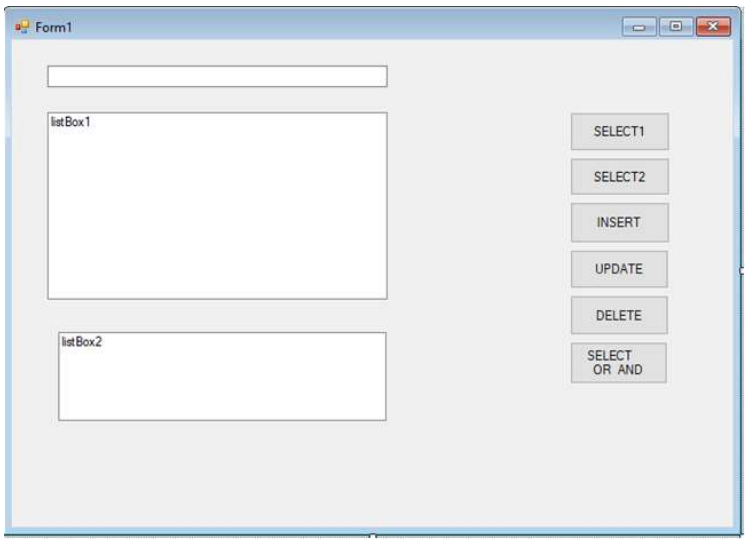



Рис. 2.7. Вікно Toolbox- панель компонентів для розробки застосування

## 2.8. Завдання до комп'ютерного практикуму

Форма є головним інструментом, за допомогою якого користувач може спілкуватися з базою даних (застосуванням). Форма може стати або прикрасою застосування, або його найневдалішою частиною. Форму можна створювати з використанням **Visual Studio.**

1. Створити проєкт DB_Test_SQL_FFF_var_N_DDDD, де FFF- прізвище студента, var_N – номер варіанта індивідуального завдання, DDDD – рік виконання комп'ютерного практикуму.
2. У Visual Studio для Windows Form Application C#. Спроектувати наступний вигляд форми.




3. Базу даних DB_Test_Sql.mdb скопіювати в папку bin/debug створеного проєкту.
4. Запрограмувати всі компоненти форми. Для цього потрібно створити відповідну компоненту, обрати необхідну подію (для кнопки – Нажатие кнопки) та в створений автоматично шаблон підпрограми скопіювати тіло підпрограми з коду, наведеного ннижче.
5. Кнопка SELECT1 в коді має ім’я button1, SELECT2 - button2, SELECT OR AND - button6, INSERT – button3, UPDATE - button4, DELETE - button5.

В створений застосуванням Visual Studio код обов'язково потрібно додати using System.Data.OleDb та підключити один з провайдерів:

***public static string ConnectString= "Provider=Microsoft.Jet.OLEDB.4.0;***
***Data Source=DB_test_Sql.mdb;";***

або

***public static string ConnectString = "Provider=Microsoft.ACE.OLEDB.12.0;***
***Data Source=DB_test_Sql.mdb;"***

для забезпечення зв’язку з таблицею бази даних ACCESS, яку ви скопіювали в папку bin/debug створеного проєкту.

Код програми, відредагований для вірного виконання наведено нижче. Звертайте увагу на коментарі в тексті коду.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data.OleDb;
```

```
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
namespace DB_Test_SQL
{
 public partial class Form1 : Form
 {
 //c# sql query select
 // public static string ConnectString = "Provider=Microsoft.Jet.OLEDB.4.0; Data Source=DB_test_Sql.mdb;";
 public static string ConnectString = "Provider=Microsoft.ACE.OLEDB.12.0; Data Source=DB_test_Sql.mdb;";
 // обов'язково потрібно додати using System.Data.OleDb;
 private OleDbConnection myConnection;
 public Form1()
 {
 InitializeComponent();
 //створюємо новий об'єкт myConnection класу OleDbConnection
 myConnection = new OleDbConnection(ConnectString);
 //встановимо зв'язок з базою даних через метод Open
 myConnection.Open();
 }
 private void button1_Click(object sender, EventArgs e)
 {
 //створимо запит, який повертатиме значення одного стовпця для одного рядка
 string query = "SELECT Фірма FROM Лічильники WHERE ((Кількість=12) OR (Кількість=22))";
 OleDbCommand command = new OleDbCommand(query, myConnection);
 textBox1.Text = command.ExecuteScalar().ToString();
```

```
}
private void button2_Click(object sender, EventArgs e)
{
//створимо запит, який повертатиме дані з декількох стовпців для декількох рядків
string query = "SELECT Кількість,Фірма,Ціна FROM Лічильники ORDER BY Фірма";
OleDbCommand command = new OleDbCommand(query, myConnection);
OleDbDataReader reader = command.ExecuteReader();
listBox1.Items.Clear();
while (reader.Read())
{
// задається шаблон виведення даних, тобто до кожного рядка буде виводитися три значення
listBox1.Items.Add(reader[0].ToString() + " " + reader[1].ToString() + " " + reader[2].ToString() + " ");
}
reader.Close();
}
private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
//обираємо у властивостях форми FormClosing і виконуємо подвійний клік
// створюється метод реагування на цю подію - буде виконуватися
//від'єднання від бази даних
myConnection.Close();
}
private void button3_Click(object sender, EventArgs e)
{
string query = "INSERT INTO Лічильники (Код_обладнання,Кількість,Фірма,Країна,Ціна)
VALUES('BTECT',45,'Cipiyc','France',888.66)";
OleDbCommand command = new OleDbCommand(query, myConnection);
```

```
command.ExecuteNonQuery();
}
private void button4_Click(object sender, EventArgs e)
{
string query = "UPDATE Лічильники SET Ціна=9999.77 WHERE Країна='Польща'";
OleDbCommand command = new OleDbCommand(query, myConnection);
command.ExecuteNonQuery();
}
private void button5_Click(object sender, EventArgs e)
{
string query = "DELETE FROM Лічильники WHERE Країна='France'";
OleDbCommand command = new OleDbCommand(query, myConnection);
command.ExecuteNonQuery();
}
private void button6_Click(object sender, EventArgs e)
{
string query = "SELECT Фірма,Кількість FROM Лічильники WHERE ((Кількість>12) AND (Кількість<70))";
OleDbCommand command = new OleDbCommand(query, myConnection);
OleDbDataReader reader = command.ExecuteReader();
listBox2.Items.Clear();
while (reader.Read())
{
// задається шаблон виведення даних, тобто до кожного рядка буде виводитися //три значення
listBox2.Items.Add(reader[0].ToString()+" "+reader[1].ToString());
}
reader.Close();
}
}
}
```

## 2.9. Контрольні запитання

1. Які ви знаєте можливості мови SQL?
2. Назвіть два основні компоненти мови SQL та їх призначення?
3. Назвіть оператори команди присутні в SQL DDL та їх призначення?
4. Назвіть оператори команди присутні в SQL DML та їх призначення?
5. Яка команда SQL видаляє запис або декілька записів у таблиці?
6. Яка команда SQL видаляє таблицю бази даних?
7. За допомогою якого метода встановлюється зв'язок з базою даних ?
8. Як реалізувати від'єднання від бази даних?
9. Поясніть призначення методу ExecuteReader().
10. Запишіть команду для виведення значень п’яти стовпчиків у listBox.

# Комп'ютерний практикум № 3. Створення інформаційної системи аналізу даних

*Мета роботи* – вивчити можливості команд мови SQL для виконання дій над даними бази даних ACCESS з інтерфейсом, розробленим у Visual Studio C#.

## 3.1. Елемент управління типу DataGridView

У Microsoft Visual Studio елемент управління dataGridView розроблений для використання в додатках, створених за шаблоном Windows Forms Application. Даний елемент управління дозволяє організовувати дані у вигляді таблиці. Дані можуть отримуватись з бази даних, колекції, внутрішніх змінних – масивів чи інших об’єктів програми [5].

Даний елемент управління можна знайти на панелі інструментів ToolBox у вкладці All Windows Forms (рис. 3.1).

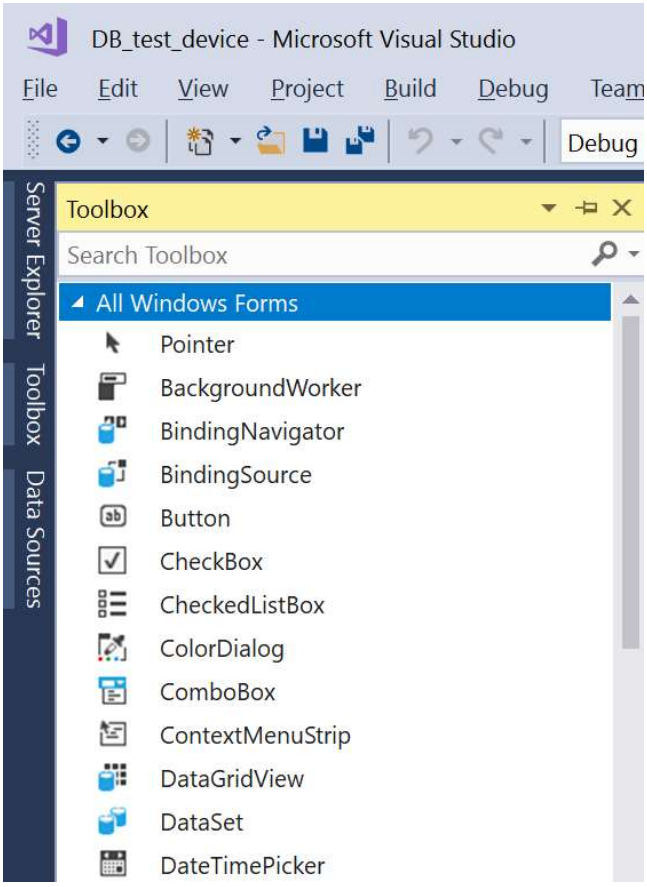



Рис. 3.1. Вікно ToolBox/All Windows Forms з елементом управління dataGridView

Після розміщення, система створює об’єкт (змінну) з іменем dataGridView1. Використовуючи ім’я елемента управління DataGridView можна використовувати методи та властивості цього елемента управління для роботи з даними.

Для того щоб реалізувати обробку даних в додатках, потрібно створити макет елемента управління DataGridView та запрограмувати відповідний алгоритм. Зовнішній вигляд елемента управління DataGridView легко налаштовується завданням значень декількох властивостей. Як джерело даних можуть використовуватися сховища даних різних типів, також елемент управління DataGridView може працювати без прив'язаного до нього джерела даних [5]. Еелемент управління DataGridView складається з стовпців і рядків, кількість яких та тип даних, які в них будуть зберігатися необхідно попередньо визначити.

Додати стовпець в dataGridView можна:

- з допомогою спеціального майстра;
- програмним шляхом.

Додавання стовпця з допомогою спеціального майстра Microsoft Visual Studio

Щоб додати стовпець в DataGridView з допомогою майстра, потрібно виконати таку послідовність дій:

- викликати меню DataGridView Tasks (рис. 3.2) шляхом кліку на стрілці вправо (права верхня частина прямокутної області dataGridView1);
- у меню DataGridView Tasks вибрати команду “Add Column…”.

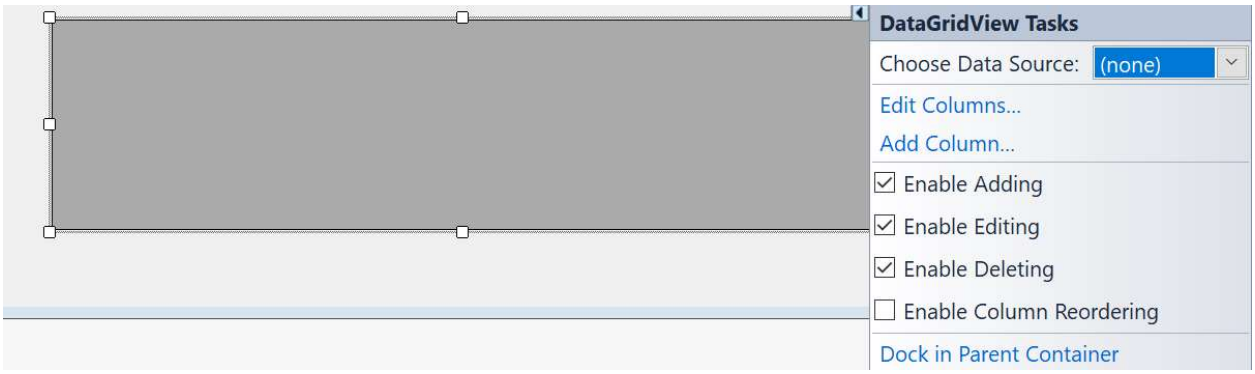



Рис. 3.2. Елемент управління dataGridView1 та вікно вибору задач DataGridView Tasks

У результаті відкриється вікно “Add Column”, в якому можна налаштувати назву колонки, тип даних колонки, назву заголовку (рис. 3.3). Для кожного стовпця можна задати властивості: Видимый, Только чтение, Зафиксирован. Після натискання кнопки Добавить у DataGridView з’являється стовпець з вибраними властивостями.

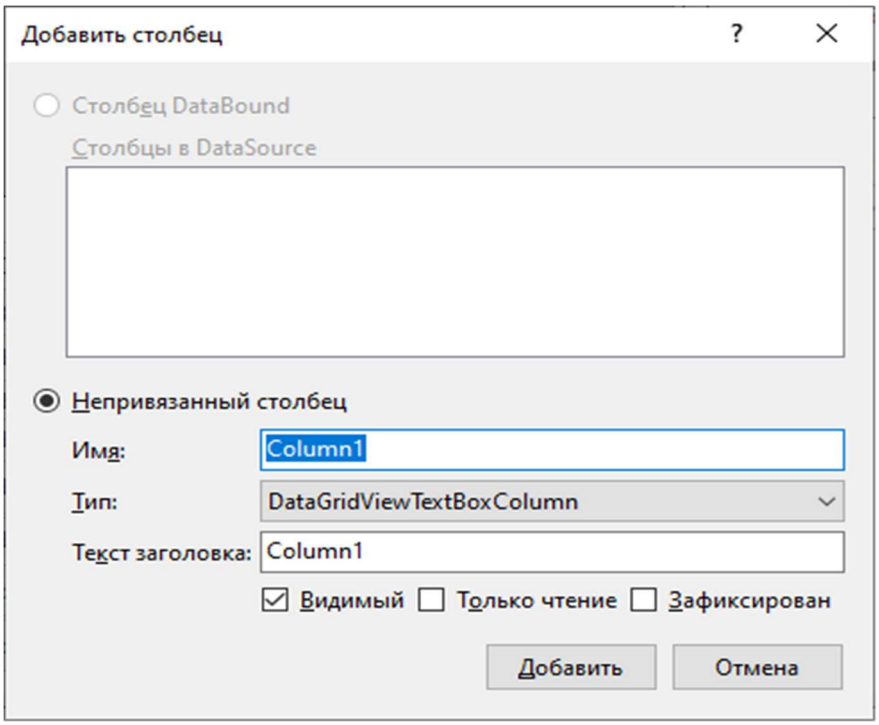



Рис. 3.3. Вікно додавання стовпця з допомогою майстра

Додавання стовпця програмним шляхом

Стовпці у dataGridView організовані у вигляді колекції Columns типу DataGridViewColumnCollection. Щоб додати стовпець програмним шляхом використовується метод (команда) Add з колекції Columns.

Метод Add має 2 варіанти реалізації:

***int DataGridViewColumnCollection.Add(DataGridViewColumn dataGridViewColumn);***

***int DataGridViewColumnCollection.Add(string ColumnName, string HeaderText);***

де DataGridViewColumn – тип System.Windows.Forms.Column що додається;

- ColumnName – назва, за якою буде здійснюватись звертання до стовпця в інших методах;
- HeaderText – Текст, що відображатиметься у заголовку колонки.

Текст обробника події додавання двох довільних стовпців наступний:

```
private void button1_Click(object sender, EventArgs e)
{
 // Додати стовпець з іменем column-1, заголовок стовпця - "Header column - 1"
 dataGridView1.Columns.Add("column-1", "Header column - 1");

 // Додати стовпець з іменем column-2
 dataGridView1.Columns.Add("column-2", "Header column - 2");

 label1.Text = "Стовпці додано";
}
```

У реальних програмах назва стовпця та його заголовку отримуються з інших елементів управління, наприклад TextBox.

Для вставки стовпця використовується метод Insert, що має таке оголошення

```
void DataGridViewColumnCollection.Insert(int columnIndex,
DataGridViewColumn dataGridViewColumn);
```

Команда Додати рядок

Додавати рядок можна двома способами:

- шляхом безпосереднього вводу з клавіатури;
- програмним шляхом.

Рядки в DataGridView організовані у вигляді колекції Rows типу dataGridViewRowCollection.

Нижче наведено обробник події, що додає 2 довільні рядки в таблицю

```
private void button3_Click(object sender, EventArgs e)
{
 // Додати рядки в таблицю
 if (dataGridView1.Columns.Count <= 0)
 {
 label1.Text = "Рядки не додано";
 return;
 }
```

```
 dataGridView1.Rows.Add("Кран", 25, "Danfoss");
 dataGridView1.Rows.Add("Фільтр", 38, "UkrSnab");
 label1.Text = "Рядки додано";
}
```

**Приклад 1.** Виведення даних до DataGridView бази даних

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data.OleDb;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
namespace Baza_dataGridView
{
 public partial class Form1 : Form
 {
 public static string ConnectString = "Provider=Microsoft.ACE.OLEDB.12.0; Data
Source=DB_test_Sql.mdb;";
 private OleDbConnection myConnection;
 public Form1()
 {
 InitializeComponent();
 LoadData();
 }
 private void LoadData()
 { //створюємо новий об'єкт myConnection класу OleDbConnection
 myConnection = new OleDbConnection(ConnectString);
//встановимо зв'язок з базою даних через метод Open
myConnection.Open();
string query = "SELECT * FROM Лічильники ORDER BY Фірма";
OleDbCommand command = new OleDbCommand(query, myConnection);
 OleDbDataReader reader = command.ExecuteReader();
```

```
List<string[]> data = new List<string[]>();
while (reader.Read())
{
// задається шаблон виведення даних, тобто до кожного рядка буде
виводитися п’ять значень
data.Add( new string[5]);
data[data.Count - 1][0] = reader[0].ToString();
data[data.Count - 1][1] = reader[1].ToString();
data[data.Count - 1][2] = reader[2].ToString();
data[data.Count - 1][3] = reader[3].ToString();
data[data.Count - 1][4] = reader[4].ToString();
}
reader.Close();
myConnection.Close();
foreach(string[] s in data)
{
dataGridView1.Rows.Add(s);
}
}
}
}
```

### 3.2. Завдання до комп'ютерного практикуму

1. Створити нову базу даних з ім’ям DB_Test_Sql_ var_N_FFF_DDDD.mdb., де FFF- прізвище студента, var_N – номер варіанта індивідуального завдання, DDDD – рік виконання комп’ютерного практикуму.
2. Імпортувати в цю базу таблицю індивідуального завдання, створену на комп’ютерному практикумі №1.
3. Створити у Visual Studio для Windows Form Application C# новий проєкт з ім’ям FFF_var_N(де FFF- прізвище студента, var_N – номер варіанта індивідуального завдання) в області Solution DB_Test_SQL_FFF_var_N_DDDD.

4. Спроектувати нову форму для виконання завдань пунктів 5 та 6.
5. Сформулювати тексти завдань для аналізу даних бази даних та реалізувати їх з використанням мови SQL:
   - чотири завдання на отримання інформації відповідно до умов пошуку команди Select.
   - по одному завданню з використанням команд Update, Delete, Insert;
6. Створити новий проєкт в
7. Для виведення результатів використати компоненти: DataGridView, ListBox, RichBox TextBox.
8. Записати до протоколу тексти завдань на обробку даних, послідовність дій по створенню інтерфейсу (компонента DataGridView), тексти SQL-команд, які необхідно виконати для отримання результату та самі результати.
9. Дати письмові відповіді на контрольні запитання.

Приклади формулювання завдань для отримання інформації з бази даних Access, з використанням мови запитів SQL та інтерфейсу, розробленому у Visual Studio C#.

1. Створити запит на вибірку до відношеня Затвори_Клапани для вибору клапанів, діаметр яких більший ніж 70 та менший за 100 і максимальна температура менша за 100. Таблиця результатів повинна складатися з полів Тип_обладнання, Діаметр, Тмакс. Впорядкування даних організуйте по зростанню значень параметра Діаметр.
2. Створити запит на вибірку до відношень Лічильники та Типи_обладнання організувати вибір лічильників, похибка вимірювання яких знаходиться у діапазоні від 1 до 3, а діаметр умовного проходу дорівнює 40 або 50.
3. Організуйте вибір інформації про комплектуючі фірми Brandoni , які зберігаються на складі і поставки яких не припинено. Таблиця результатів повинна складатися з полів Тип_обладнання, Виробник, Поставки_припинено.
4. Організувати вибір клапанів фірм Hans Sasserath та Danfoss, ціна яких не перевищує 80 грн.,а кількість наявних на складі дорівнює кількості

мінімального запасу. Таблиця результатів повинна складатися з полів Тип_обладнання, Виробник, Ціна, На_складі.

***5.*** Створити запит на видалення до відношення Обладнання_нова. Організуйте видалення даних про комлектуючі, кількість яких на складі дорівнює мінімальному запасу . Таблиця результатів повинна складатися з полів Ціна, На_складі, Очікується, Мін_запас.

***6.*** Створити запит для оновлення ціни комплектуючих (ціну зменшити на 20%) , наявність яких на складі менша ніж на 10 за можливий мінімальний запас.

***7.*** Створити запит для розв’язання наступного завдання: якщо поставки припинено і на складі залишилося менше ніж 10 виробів, то зменшити ціну на 50%, інакше ціну залишити незмінною.

### 3.3. Контрольні запитання

1. Яка команда SQL видаляє таблицю бази даних?
2. За допомогою якого метода можна програмно видалити стовпчик DataGridView?
3. За допомогою якого метода можна програмно видалити рядок DataGridView??
4. Назвіть призначення параметра ORDER BYкоманди SELECT?
5. Назвіть призначення параметра GROUP BYкоманди SELECT?
6. Яка команда SQL використовується для додавання нових даних до таблиці бази даних?
7. За допомогою якого метода встановлюється зв'язок з базою даних ?
8. Як реалізувати виведення даних до елемента керування DataGridView?
9. Поясніть призначення класу OleDbDataReader.
10. Яка команда SQL оновлює дані в записах таблиці?

# Комп'ютерний практикум № 4. Створення простих запитів в СУБД ACCESS

*Мета роботи* – вивчити можливості конструктора запитів для створення таблиць, побудованих з використанням даних з основних таблиць.

## 4.1. Теоретичні відомості

В СУБД Access виконувати різноманітні дії над даними можна використовуючи об’єкт Запит.

Запити є робочими інструментами баз даних ACCESS, за допомогою яких користувач має можливість отримувати необхідну інформацію з бази даних. В СУБД ACCESS реалізовані наступні типи запитів:

- запит на простий вибір даних;
- запит на оновлення даних;
- запит на додавання даних;
- запит на об′єднання даних;
- запит на видалення даних;
- перехресний запит;
- параметричний запит;
- підсумковий запит;
- запит до сервера.

Для того, щоб переглянути список доступних запитів відповідної версії СУБД Access необхідно вибрати опції головного меню Создание/Конструктор запросов, після чого відкриється вікно з піктограмами вбудованих запитів та бланком шаблона критерія для створення нового запиту (рис. 4.1).

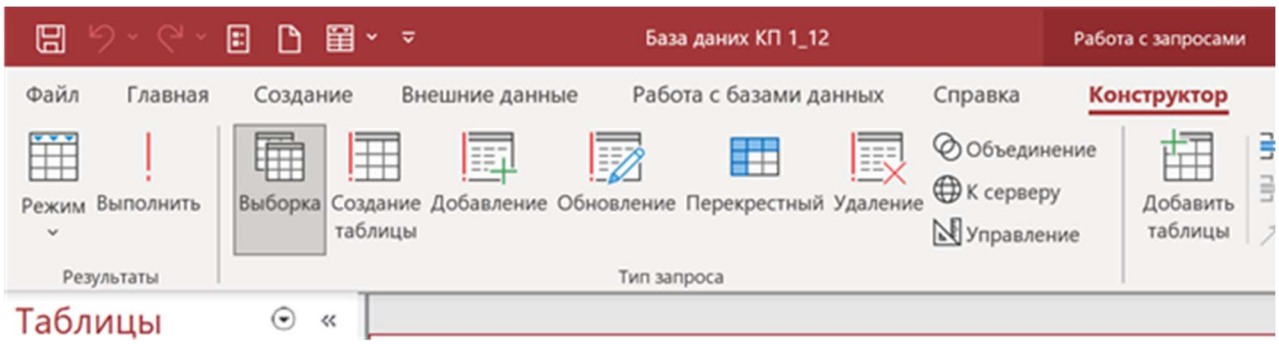



Рис. 4.1. Вікно з піктограмами вбудованих запитів СУБД ACCESS

В СУБД Access запити можна будувати за допомогою операторів мови структурованих запитів SQL (Structured Query Language), або за допомогою візуальних засобів QBE Grid (Query-By-Example), який представляє собою конструктор запитів.

В СУБД Access існує два автоматизовані способи створення запитів: майстер запитів та конструктор запитів. Перший спосіб дає можливість створювати обмежену кількість запитів з використанням наявних шаблонів. Другий спосіб дає можливість створювати унікальні запити для розв'язку нестандартних завдань.

За допомогою запитів можна:

• переглянути дані з баз даних, отримані у відповідності до сформульованих умов, у вигляді віртуальної таблиці або запам'ятати отриману інформацію у новоствореній таблиці;

• видалити дані, заданих у відповідності до сформульованих умов, рядків таблиці;

• оновити дані стовпців таблиці;

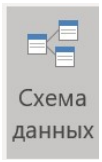
• організувати пошук даних у відповідності до значення заданого параметра

• виконати групування даних та обчислення із згрупованими даними.

Результат запитів можна зберігати у файлах баз даних, а можна запускати на виконання і лише переглядати результати виконання умов запиту.

## 4.2. Зв'язки між таблицями бази даних

Для отримання інформації з бази даних з використанням запитів найчастіше використовуються декілька таблиць. Тому перед створенням запитів, необхідно перевірити чи вірно встановлені зв'язки між таблицями. Використовуючи опцію головного меню СУБД Access Работа с базами данных/ Схема данных можна переглянути встановлені зв'язки між таблицями бази даних. Кнопка Схема данных на панелі інструментів має наступний вигляд




(рис. 4.2). Якщо зв’язки між таблицями вже були встановлені, то вікно із схемою даних необхідно закрити. Якщо ж у базі даних зв’язки не були визначені або додані нові таблиці, то необхідно відкрити вікно **Добавление таблиц** (рис. 4.3) та встановити зв’язки.

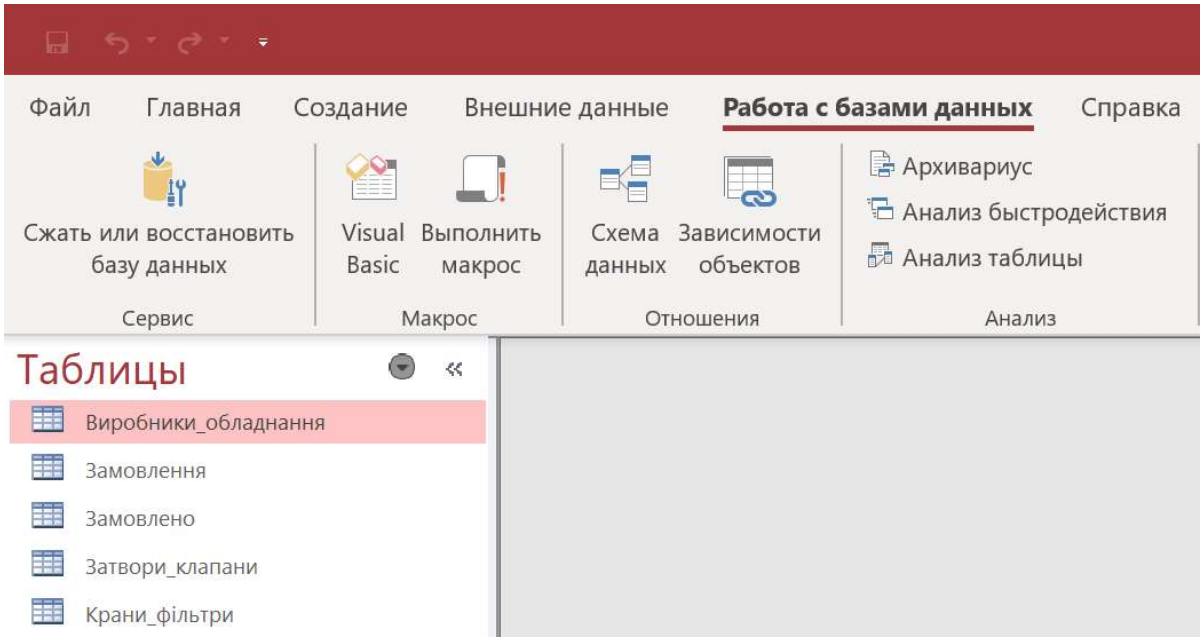



Рис. 4.2. Вікно з меню опції **Работа с базами данных** для перегляду зв’язків між таблицями бази даних

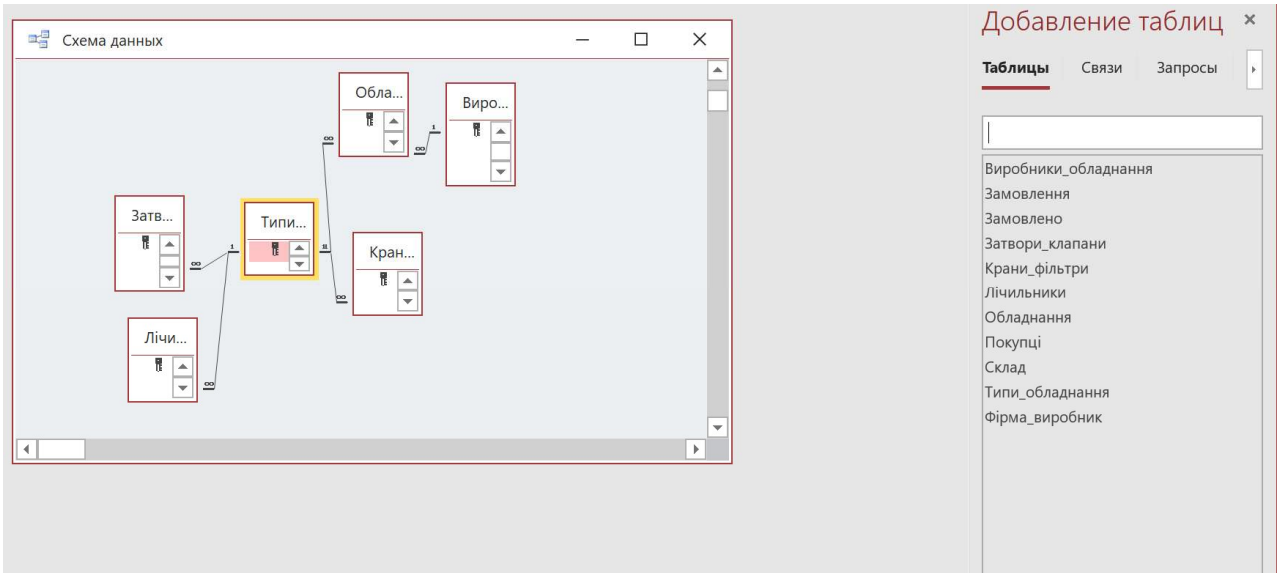



Рис. 4.3. Вікно конструктора для перегляду зв’язків між таблицями бази даних або встановлення нових зв’язків

Виконати подвійний клік мишою на іменах таблиць, які потрібно використати для створення запиту, після чого закрити активне діалогове вікно.

Якщо ж зв'язок між таблицями попередньо не були встановлені, то потрібно вибрати поле в одній таблиці і перетягнути його на відповідне поле іншої таблиці (зв'язані поля не обов'язково повинні мати однакові імена, але вони повинні мати однакові типи даних і мати вміст одного типу). Відкриється вікно **Изменение связей** (рис. 4.4).

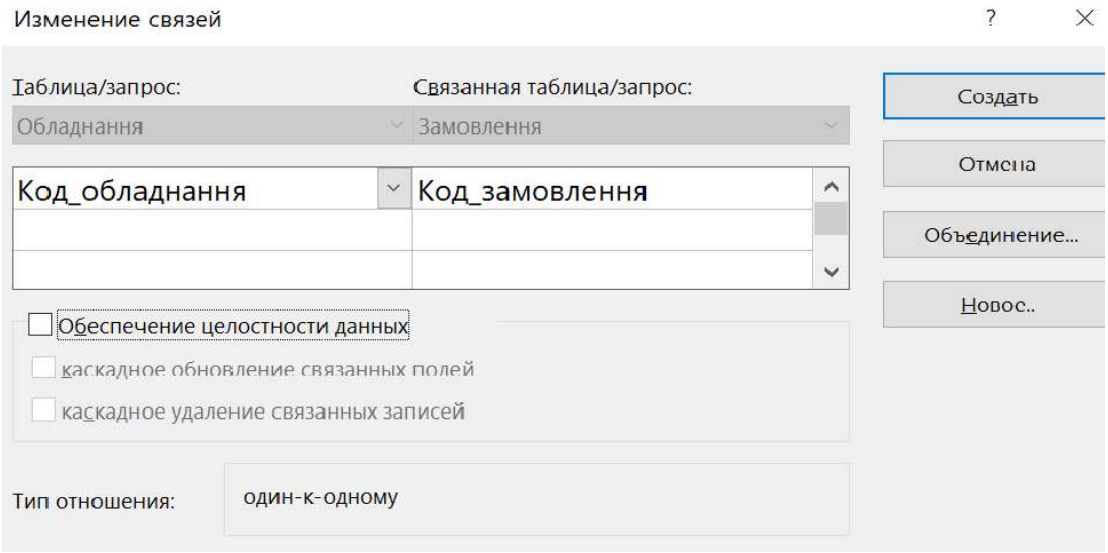



Рис. 4.4. Вікно **Изменение связей** для встановлення зв'язків між таблицями

Для створення вірного зв'язку потрібно вибрати параметри:

- Обеспечение целостности данных;
- Каскадное обновление связанных полей;
- Каскадное удаление связанных записей

та натиснути кнопку **Создать.** Після закриття вікна **Схема данных** программа запитає чи потрібно запам'ятати макет. Задані зв'язки буде запам'ятовано у базі даних.

## 4.3. Створення запиту за допомогою конструктора

Створення запиту починається з вибору в головному меню опції **Создание /Конструктор запросов**. Після цього відкриється вікно з бланком шаблона критерія для створення запиту (рис. 4.5).

З вікна **Добавление таблиц** необхідно вибрати одну або декілька таблиць та для обраних полів таблиць записати умови вибору (рис. 4.5).

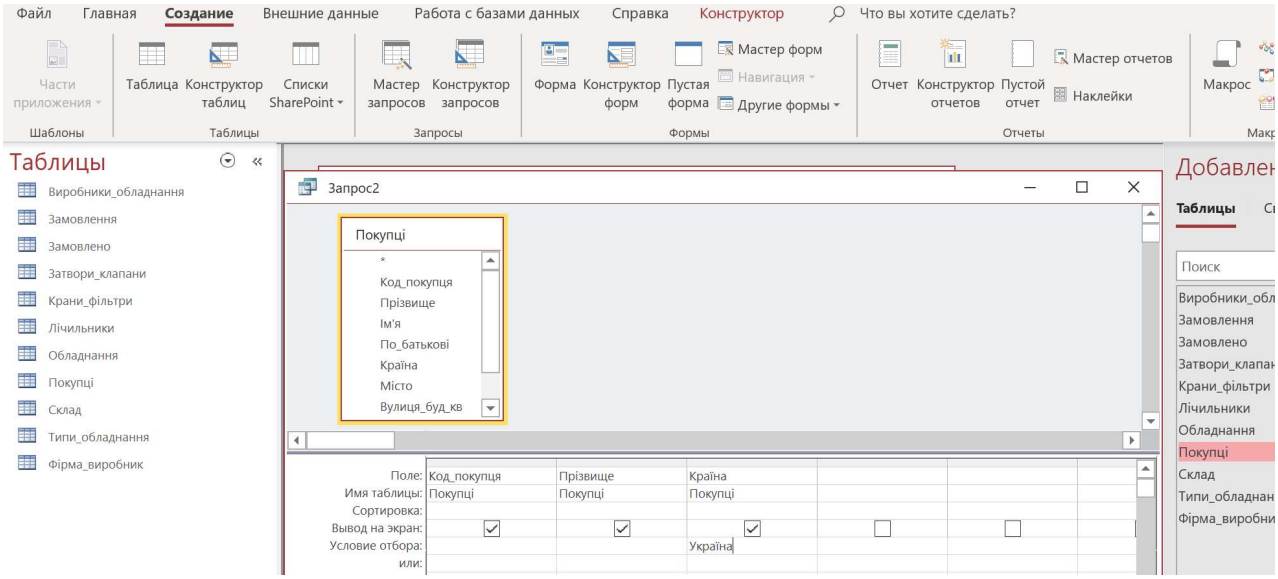



Рис. 4.5. Вікно конструктора для створення запитів

Бланк шаблона конструктора запитів використовується для створення запиту. У кожний стовпчик бланка запиту вводяться назви полів та імена таблиць, яким вони належать. Далі можна вказати дані яких полів мають бути виведені у результуючу таблицю та режим сортування даних, якщо в цьому є потреба. В рядок **Условие отбора** записується умова для вибору даних з відповідних таблиць. Якщо умова не вказана у результуючу таблицю будуть виведені дані всіх полів, що вказані у бланку запиту. В рядку **Вывод на экран** шаблона конструктора відмічені поля, значення яких має бути виведено до результуючої таблиці. Для цього запиту до результуючої таблиці будуть виведені для покупців з України коди покупців та їх прізвища.

Після закінчення введення всіх необхідних параметрів та набору умов у бланк запиту, можна переглянути результати його виконання, для цього необхідно натиснути кнопку **Выполнить**  на панелі інструментів. Якщо отримані результати відповідають поставленій задачі, то запит необхідно зберегти, надавши ім’я новоствореному запиту, та запустити запит на виконання.

Умови для декількох полів рядка **Условие отбора** будуть зв'язані між собою логічним оператором **AND**. Умови для полів рядка **Условие отбора** та **или** будуть зв'язані між собою логічним оператором **OR**. Для кожного поля можна записати складну умову, використовуючи логічні операції відношення або логічні оператори. Наприклад, для поля Країна таблиці Покупці можна записати умову "Україна" **OR** "Польща". За такої умови до результуючої таблиці будуть виведені для покупців з України та Польщі коди покупців та їх прізвища. Більше варіантів логічних виразів буде розглянуто у прикладах нижче.

Для перегляду запитів в Access існує три режими (рис. 4.6):

- Режим таблицы, який дозволяє переглянути результати запиту без запуску його на виконання;
- Режим SQL, який дозволяє переглянути команду SQL, що була створена у фоновому режимі;
- Конструктор, цей режим повертає до бланку критерія шаблону.

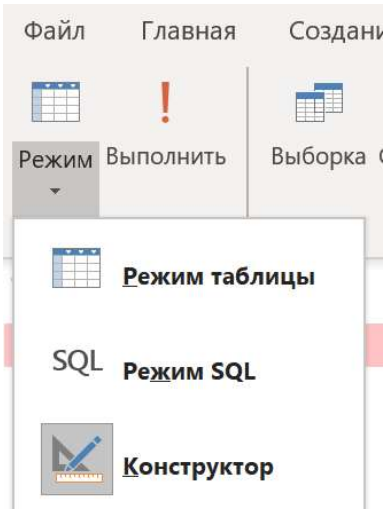



Рис. 4.6. Режими перегляду результатів запиту до його виконання, команди SQL та повернення в режим конструктора

**Приклад 4.1.** У вікні запиту **Выборка** (рис. 4.7) наведено приклад запису запиту на вибірку у режимі Конструктор. Результатом виконання запиту буде таблиця даних, вибраних з полів таблиць **Обладнання і Типи обладнання** (рис. 4.8). На рис. 4.8 наведено лише частину даних.

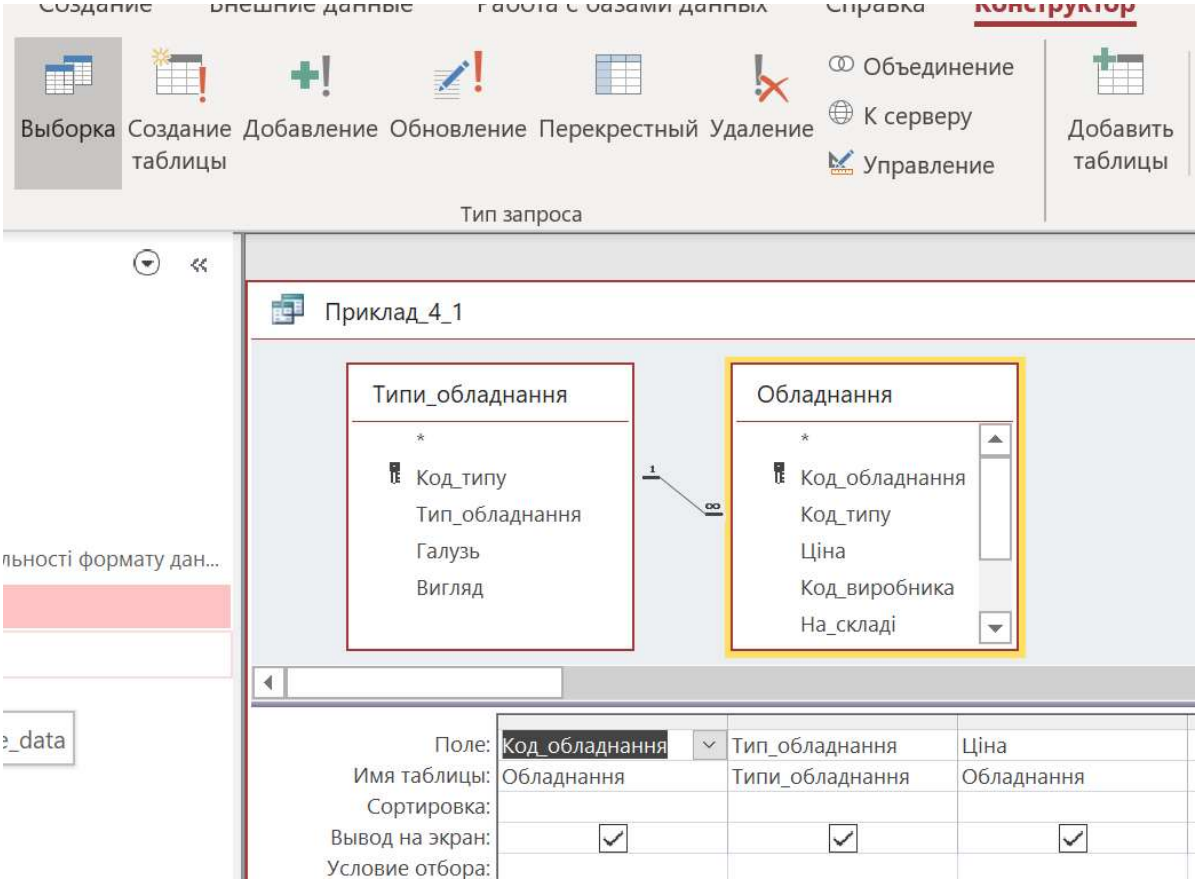



Рис. 4.7. Вигляд запиту для приклада 4.1 в режимі конструктора

Запит у вигляді SQL- рядка матиме наступний вигляд для приклада 4.1:

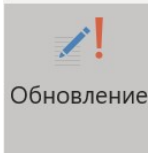
***SELECT Обладнання.Код_обладнання, Типи_обладнання.Тип_обладнання, Обладнання.Ціна***
***FROM Типи_обладнання INNER JOIN Обладнання ON***
***Типи_обладнання.Код_типу = Обладнання.Код_типу;***

Приклад_4_1

| Код_обладн | Тип обладнання | Ціна |
|---|---|---|
| J900001 | Затвір поворотний | 187,92 грн. |
| J900002 | Затвір поворотний | 197,16 грн. |
| J900003 | Затвір поворотний | 197,16 грн. |
| J900004 | Затвір поворотний | 210,96 грн. |
| J900005 | Затвір поворотний | 231,48 грн. |
| J900006 | Затвір поворотний | 265,92 грн. |
| J900007 | Затвір поворотний | 339,24 грн. |
| J900008 | Затвір поворотний | 418,80 грн. |
| J900009 | Затвір поворотний | 541,20 грн. |
| J900010 | Затвір поворотний | 807,00 грн. |
| J900011 | Затвір поворотний | 404,00 грн. |

Рис. 4.8. Результат виконання запиту для приклада 4.1

## 4.4. Заміна групи записів за допомогою запиту на оновлення

Запит, що вносить зміни , можна розглядати як звичайний запит, який повинен виконувати деякі операції над вказаною групою записів у результуючій таблиці. У якості джерела даних можуть бути вибрані таблиці або запити, які містять записи, дані яких необхідно оновити, та поля, які будуть використані у критерії відбору. Для створення такого запиту доцільно створити спочатку запит на вибірку, перевірити результати його виконання. Якщо результати виконання запиту на вибірку нас задовольняють, можна перетворити цей запит на запит оновлення , залишивши критерії вибору тими самими. Для цього у режимі конструктора запиту **Создание /Конструктор запросов** виберіть на панелі інструментів команду **Обновление** 

.

Перетягнути із списку полів у бланк критерія відбору поле, яке має бути оновленим або використаним в умовах відбору. У полі **Обновление** записуємо вираз, результат якого буде записано у відповідне поле замість старих значень. Вираз може бути записаним безпосередньо у полі критерія відбору **Обновление** , або для запису виразу можна використати **Построитель** 

. У разі необхідності можна задати умови відбору в полі **Условие отбора** .

Для того щоб переглянути список полів записів, значення яких будуть оновлені, натисніть кнопку Вид (Режим таблицы)  на панелі інструментів. Список, що виводиться буде містити нові значення, які будуть запам’ятовані після запуску запиту на виконання. Для повернення до режиму конструктора запиту натисніть кнопку Вид (Конструктор)  на панели інструментів та запам’ятайте створений запит та натиснути кнопку **Выполнить**  на панелі інструментів, щоб оновити записи.

***Важливо!!!*** Для виконання прикладів 4.2 та 4.3 створіть таблицю Копія_Склад_обладнання і виконайте запити до цієї таблиці замість таблиці Склад_обладнання.

**Приклад 4.2.** Зменшити ціну лічильників на 20%, якщо значення діаметра умовного проходу у них дорівнює 80.

Для розв'язання поставленої задачі необхідно вибрати таблиці **Склад_обладнання та Лічильники_запити**. Для поля Ціна, значення якого мають бути оновленими, введіть у комірку **Обновление** вираз [Обладнання]![Ціна]*0,8. В поле **Условие отбора** необхідно для параметра **Діаметр_умовного_проходу** задати умову відбору - 80. Далі потрібно переглянути результати виконання запиту і якщо вони вірні – запустити запит на виконання. Після цього старі значення у відповідному полі таблиці буде замінено новими значеннями.

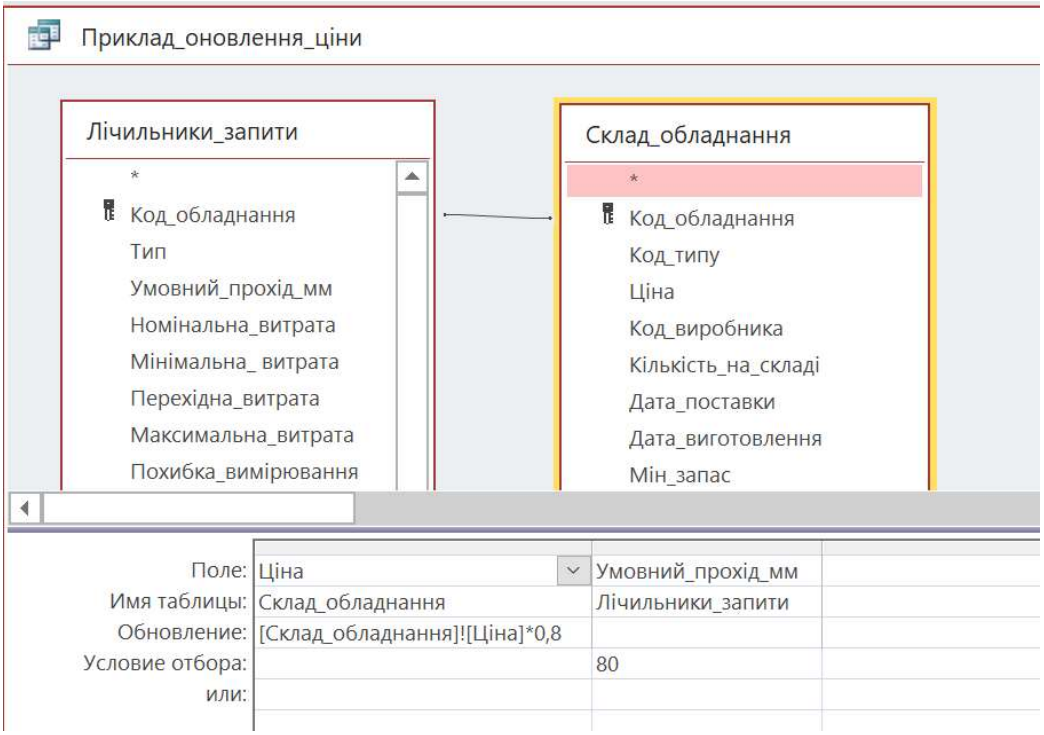



Рис. 4.9. Вигляд запиту на оновлення у конструкторі запитів

Запит у вигляді SQL- рядка повинен мати наступний вигляд для приклада :

***UPDATE Склад_обладнання INNER JOIN Лічильники_запити ON Склад_обладнання.Код_обладнання = Лічильники_запити.Код_обладнання***
***SET Склад_обладнання.Ціна =[Склад_обладнання]![Ціна]*0.8***
***WHERE (((Лічильники_запити.Умовний_прохід_мм)=80));***

Наступні два приклади демонструють отримання різних результатів після виконання команд на оновлення. В результаті виконання запиту значення поля Кількість_на_складі, які дорівнюють 2 будуть замінені на значення 20.

***UPDATE Обладнання_add SET Склад_обладнання _add.Кількість_на_складі = +20***

***WHERE (((Склад_обладнання _add.Кількість_на_складі)=2));***

В результаті виконання запиту до значення поля Кількість_на_складі буде додано 20 нових одиниць обладнання , якщо значення поля Мін_запас менше за 5

***UPDATE Склад_обладнання_add SET***

***Склад_обладнання_add.Кількість_на_складі = [Склад_обладнання _add]![Кількість_на_складі]+20***

***WHERE ((([Склад_обладнання_add]![ Кількість_на_складі]-***

***[Склад_обладнання_add]![Мін_запас])<5));***

Виконати два останні приклади. Для цього скопіювати таблицю Склад_обладнання та зберегти її з ім’ям Склад_обладнання_add. Проаналізувати отримані результати. Зберегти запити з іменами Приклад_4_2_а та Приклад_4_2_б.

## 4.5. Створення запиту на видалення

З усіх запитів, що вносять зміни, запит на видалення найбільш небезпечний. Цей тип запиту видаляє записи з таблиці назавжди без можливості відмінити результат видалення. Як і всі інші запити, що вносять зміни, запит на видалення обробляє групу записів у відповідності до критерію вибору. При видаленні записів із зв’язаних таблиць, що перебувають у відношенні один-до-багатьох, необхідно перевірити чи включена опція **Каскадное обновление**. Запит на видалення видаляє записи повністю. Якщо ж потрібно видалити дані у окремих полях – необхідно використати запит на оновлення. Для створення запиту на видалення потрібно створити новий запит в режимі конструктора та вибрати опції головного меню **Запрос** – **Удаление** . У бланку запиту з’явився новий рядок Удаление . Цей рядок може приймати значення одного з параметрів списку **Из** або **Условие.** Параметр **Из** вказує з якої таблиці та які поля потрібно видалити, а параметр **Условие** вказує за виконання якої умови видалення має відбутися.

**Приклад 4.3.** Застосовуючи мову запитів **SQL** до відношення **Склад_обладнання** організуйте видалення даних про комлектуючі, мінімальний запас яких на складі дорівнює 0. Таблиця результатів повинна складатися з полів Код_обладнання, Код_типу, Ціна, Кількість_на_складі.

Для розв'язання поставленої задачі необхідно скопіювати таблицю **Склад_обладнання** та зберегти її з ім'ям **Склад_обладнання_delete**. Вибрати таблицю **Склад_обладнання_delete**. Створити новий запит для цієї таблиці та та вибрати опції головного меню **Запрос** – **Удаление.** У бланку критерія запиту у рядку Удаление вибираємо параметр **Условие.** В рядку Условие отбора задаємо полю **Дата_виготовлення** значення #16.12.2021#, як показано на рис.4.10. Це означає, що будуть видалені всі записи з таблиці **Склад_обладнання_delete**, для яких значення поля **Дата_виготовлення** дорівнює 16.12.2021. Запускаємо запит на виконання. Далі потрібно переглянути результати виконання запиту і якщо вони вірні – запустити запит на виконання. Після цього рядки, значення поля **Дата_виготовлення** яких відповідає заданій умові, буде видалено.

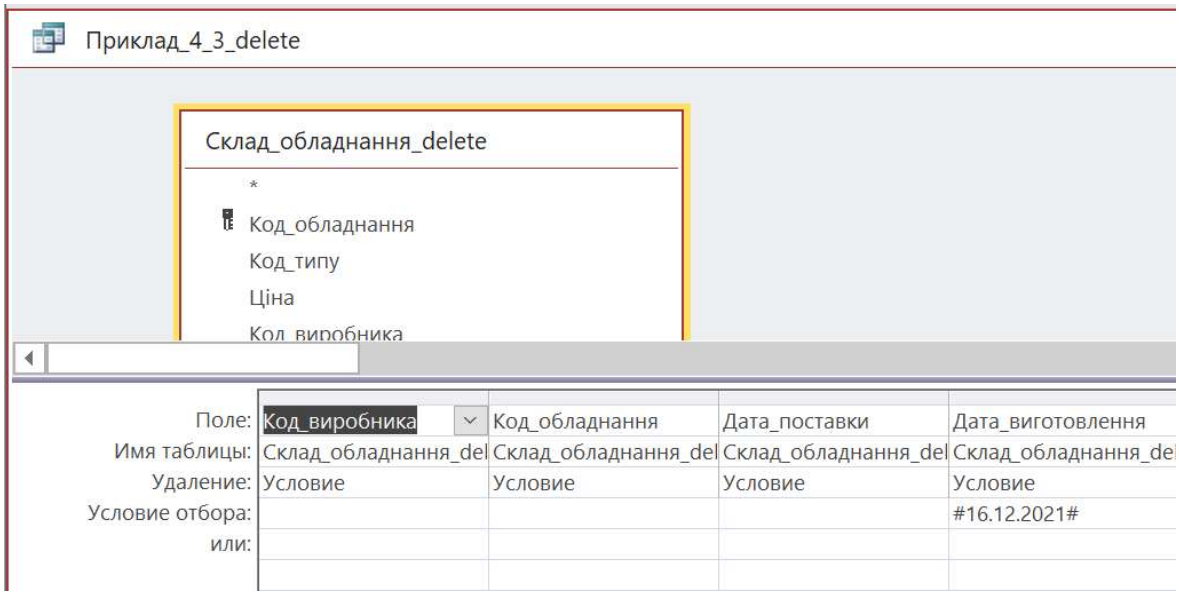



Рис. 4.10 Вигляд запиту на видалення у конструкторі запитів

Запит у вигляді SQL- рядка для приклада 4.3 повинен мати наступний вигляд:

***DELETE Склад_обладнання_delete.Код_виробника,***
***Склад_обладнання_delete.Код_обладнання,***

***Склад_обладнання_delete.Дата_поставки,***
***Склад_обладнання_delete.Дата_виготовлення***
***FROM Склад_обладнання_delete***
***WHERE (((Склад_обладнання_delete.Дата_виготовлення)=#12/16/2021#));***

## 4.6. Завдання до комп'ютерного практикуму

1. Імпортувати до своєї бази, створеної на комп'ютерному практикумі №1, з бази даних db_stud_comp_prak_4.accdd таблиці: Лічильники_газу, Лічильники_запити, Крани_Фільтри, Затвори_Клапани.
2. Відредагувати схему бази даних, встановивши зв’язки з імпортованими таблицями.
3. Виконати всі приклади, наведені у теоретичних відомостях та зберегти їх з іменами, наприклад, Test_4_1_FFF. Виконати завдання зберегти їх з іменами, наприклад, Query_4_1_FFF. Де FFF - прізвище студента. Записати до протоколу послідовність дій(опції меню, критерій конструктора запитів та текст SQL-команди), які необхідно виконати для отримання результату та самі результати. Приклад оформлення результатів комп'ютерного практикуму №4 представлено у додатку **Д.1.**
4. Завдання для виконання запитів.

4.1. Створити запит на вибірку, до відношень Назва_обладнання та Крани_Фільтри для вибору кранів з умовним проходом більшим ніж 15 та меншим за 50, умовний тиск яких дорівнює 69. Таблиця результатів повинна складатися з полів Тип_обладнання, Умовний_прохід, Умовний_тиск.

4.2. Створити запит на вибірку до відношення Затвори_Клапани для вибору клапанів, діаметр яких більший ніж 70 та менший за 100 і максимальна температура менша за 100. Таблиця результатів повинна складатися з полів Тип_обладнання, Діаметр, Тмакс. Впорядкування даних організуйте по зростанню значень параметра Діаметр.

4.3. Створити запит на вибірку до відношень Лічильники_запити та Назва_обладнання організувати вибір лічильників, похибка вимірювання яких знаходиться у діапазоні від 1 до 3, а діаметр умовного проходу дорівнює 40 або 50.

4.4. Організуйте вибір інформації про комплектуючі фірми Brandoni , які зберігаються на складі і поставки яких не припинено. Таблиця результатів повинна складатися з полів Тип_обладнання, Виробник, Поставки_припинено.

4.5. Організувати вибір клапанів фірм Hans Sasserath та Danfoss, ціна яких не перевищує 80 грн.,а кількість наявних на складі дорівнює кількості мінімального запасу. Таблиця результатів повинна складатися з полів Тип_обладнання, Виробник, Ціна, Кількість_на_складі.

4.6. Скопіювати таблицю Склад_обладнання та надати їй ім’я Склад_обладнання_нова. Створити запит на видалення до відношення Склад_обладнання_нова. Організуйте видалення даних про комлектуючі, кількість яких на складі дорівнює мінімальному запасу . Таблиця результатів повинна складатися з полів Ціна, Кількість_на_складі, Дата_виготовлення, Мін_запас.

4.7. Скопіювати таблицю Склад_обладнання та надати їй ім’я Склад_обладнання_оновлено. Створити запит для оновлення ціни комплектуючих (ціну зменшити на 20%) , наявність яких на складі менша за можливий мінімальний запас.

4.8. Скопіювати таблицю Склад_обладнання та надати їй ім’я Склад_обладнання_ціна. Створити запит для розв’язання наступного завдання: якщо поставки припинено і на складі залишилося менше ніж 10 виробів, то зменшити ціну на 50%, інакше ціну залишити незмінною.

4.9. Скопіювати таблицю Склад_обладнання та надати їй ім’я Склад_обладнання_додано. Збільшити кількість товарів **(**відношення Склад_обладнання_додано**)**, що зберігаються на складі (Поле Кількість_на_складі) на 50, якщо залишок цього товару на складі складає менше ніж 5 одиниць за мінімально можливий запас. Таблиця результатів повинна складатися з полів Кількість_на_складі, Мін_запас.

5. Дайте письмові відповіді на контрольні запитання.

## 4.7. Контрольні запитання

1. Які режими виконання запитів SQL існють в ACCESS?
2. За допомогою яких операторів виконується маніпулювання даними?
3. Для чого в операторі SELECT використовується параметр ORDER BY?
4. Для чого використовується параметр GROUP BY?
5. Назвіть п'ять узагальнюючих функцій, які використовуються при створенні запитів?

### Додаток

Результати виконання прикладів і завдань повинні мати такий вигляд:

текст завдання, запит в режимі конструктора, запит в режимі SQL, результат виконання запиту приклада чи індивідуального завдання. Наприклад:

**Приклад 4.16.** Створити запит на вибірку для вибору ціни клапанів. Таблиця результатів повинна складатися з полів Код_обладнання, Тип_обладнання, Ціна.

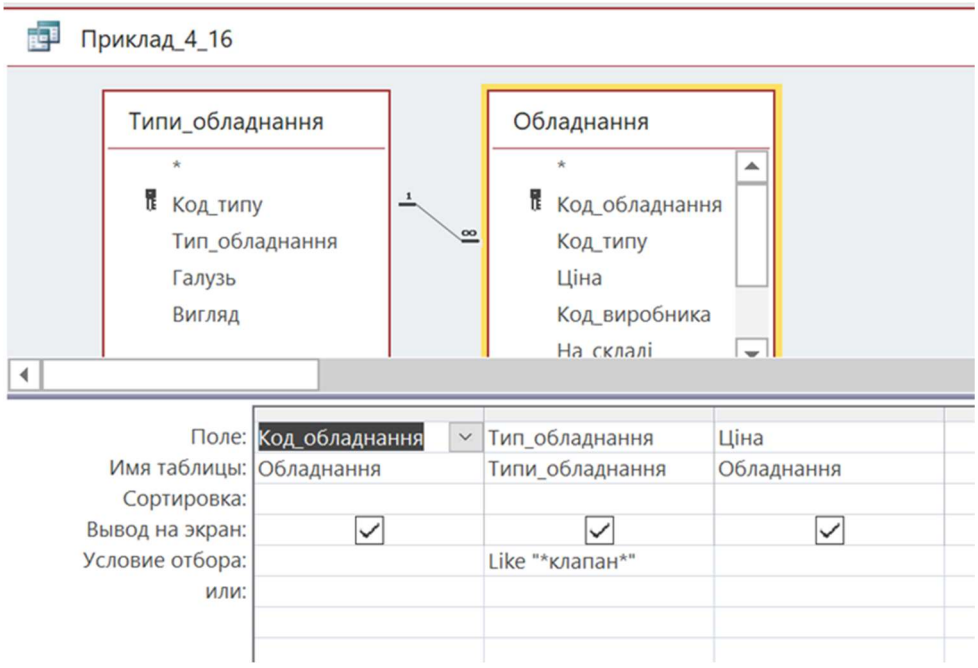



Рис. Д.1 Запит в режимі конструктора

***SELECT Обладнання.Код_обладнання, Типи_обладнання.Тип_обладнання, Обладнання.Ціна***

***FROM Типи_обладнання INNER JOIN Обладнання ON Типи_обладнання.Код_типу = Обладнання.Код_типу WHERE (((Типи_обладнання.Тип_обладнання) Like "*клапан*"));***

***Рисунок Д.2 Запит в режимі SQL***

Приклад_4_16

| Код_обладн | Тип обладнання | Ціна |
|---|---|---|
| 202bz_15 | Балансировочний клапан | 223,74 грн. |
| 202bz_20 | Балансировочний клапан | 238,92 грн. |
| 202bz_25 | Балансировочний клапан | 265,32 грн. |
| 202bz_32 | Балансировочний клапан | 295,02 грн. |
| 202bz_40 | Балансировочний клапан | 331,20 грн. |
| 202bz_50 | Балансировочний клапан | 396,00 грн. |
| 202zz_15 | Балансировочний клапан | 276,54 грн. |
| 202zz_20 | Балансировочний клапан | 291,72 грн. |
| 202zz_25 | Балансировочний клапан | 318,12 грн. |
| 202zz_32 | Балансировочний клапан | 347,82 грн. |
| 202zz_40 | Балансировочний клапан | 384,00 грн. |
| 202zz_50 | Балансировочний клапан | 448,80 грн. |
| 445_100 | Балансировочний клапан | 267,20 грн. |
| 445_65 | Балансировочний клапан | 63,00 грн. |
| 445_80 | Балансировочний клапан | 98,24 грн. |
| 1915_13_03 | Запобіжний клапан | 51,00 грн. |
| 1915_13_06 | Запобіжний клапан | 51,00 грн. |

Рис. Д.3. Результат виконання запиту приклада 4.16

# Комп'ютерний практикум № 5. Створення складних запитів в СУБД ACCESS

*Мета роботи* – вивчити можливості конструктора запитів для створення складних запитів з використанням декількох зв'язаних таблиць, агрегуючих функцій та функцій для обробки даних типу Дата/Время.

## 5.1. Складні запити

До складних запитів можна віднести запити, що використовують узагальнючі функції, запити з підзапитами та перераховані нижче запити:

- запит на додавання даних;
- запит на об′єднання даних;
- перехресний запит;
- параметричний запит.

При конструюванні таких запитів потрібно використовувати допоміжні функції або допоміжні запити. В цих запитах можуть використовуватися інші команди мови SQL при записі умов для вибору необхідної інформації.

Узагальнюючі функці

Стандарт ISO містить визначення наступних **узагальнюючих (агрегуючих)** функцій**,** які виконуються над згрупованими записами.

Таблиця 5.1. Узагальнюючі функції

| п/п | Функція | Результат |
|---|---|---|
| 1 | COUNT | Повертає кількість значень у вказаному стовпці |
| 2 | SUM | Повертає суму значень у вказаному стовпці |
| 3 | AVG | Повертає усереднене значення у вказаному стовпці |
| 4 | MIN | Повертає мінімальне значення у вказаному стовпці |
| 5 | MAX | Повертає максимальне значення у вказаному стовпці |
| 6 | StDev | Повертає середньоквадратичне відхилення від середнього значення поля у групі; |
| | Var | Повертає дисперсію значень поля у групі; |
| | First, Last | Повертає значення поля з першого й останнього запису в групі. |

Всі ці функції оперують із значеннями в одному стовпці таблиці і повертають по одному запису для кожної групи. До запиту, як правило, включаються поля, по яким було виконано групування, та поля, до яких застосовуються агрегуючі функції, або вводяться умови відбору записів перед виконанням процедури групування.

Функції COUNT, MIN і MAX застосовуються як до числових, так і до нечислових полів, тоді як функції SUM і AVG можуть використовуватися тільки з числовими полями. Винятком є функція COUNT(*), при обчисленні результатів будь-якої функції спочатку виключаються всі порожні значення, після чого необхідна операція застосовується тільки до конкретних значень стовпця, що залишилися. Варіант COUNT (*) є особливим випадком використання функції COUNT — його призначення полягає в підрахунку всіх рядків таблиці, незалежно від того, містяться там порожні значення, або значення, що дублюються або будь-які інші значення.

## 5.2. Запити з підзапитами

Розглянемо використовування операторів SELECT, вбудованих в тіло іншого оператора SELECT. **Зовнішній** (другий) оператор SELECT використовує результат виконання **внутрішнього** (першого) оператора для визначення змісту остаточного результату всієї операції. Внутрішні запити можуть бути розміщені в параметрі WHERE і HAVING зовнішнього оператора SELECT — в цьому випадку вони одержують назву **підзапитів,** або **вкладених запитів.** Крім того, внутрішні оператори SELECT можуть використовуватися в операторах INSERT, UPDATE і DELETE. Існує три типи підзапитів.

- *Скалярний підзапит* повертає значення, вибране з перетину одного стовпця з одним рядком, — тобто єдине значення. У принципі, скалярний підзапит може використовуватися скрізь, де вимагається вказати єдине значення.
- *Рядковий підзапит* повертає значення декількох стовпців таблиці, але у вигляді єдиного рядка. Рядковий підзапит може використовуватися

скрізь, де застосовується конструктор рядкових значень — звичайно це предикати.

- *Табличний підзапит* повертає значення одного або більше стовпців таблиці, розміщені в більш ніж одному рядку. Табличний підзапит може використовуватися скрізь, де допускається указувати таблицю — наприклад, як операнд предиката IN.

Підзапит є інструментом створення тимчасової таблиці, вміст якої вибирається і обробляється зовнішнім оператором. Підзапит може вказуватися безпосередньо після операторів порівняння (тобто операторів = < > <= >= <>) в параметрах WHERE і HAVING. Текст підзапиту повинен бути укладений в дужки.

До підзапитів застосовуються наступні правила і обмеження:

1. В підзапитах не повинен використовуватися параметр ORDER BY, хоча він може бути присутнім у зовнішньому запиті.

2. Список в команді SELECT підзапиту повинен складатися з імен окремих стовпців або складених з них виразів — за винятком випадку, коли в підзапиті використовується ключове слово EXISTS.

3. За замовчуванням імена стовпців в підзапиті відносяться до таблиці, ім'я якої вказано в його параметрі FROM. Проте допускається посилатися і на стовпці таблиці, вказані у параметрі FROM зовнішнього запиту, для чого використовуються кваліфіковані імена стовпців.

4. Якщо підзапит є одним з двох операндів, що беруть участь в операції порівняння, то запит повинен указуватися в правій частині цієї операції.

Ключові слова ANY і ALL можуть використовуватися з підзапитами, що повертають один стовпець чисел. Якщо підзапиту передуватиме ключове слово ALL, умова порівняння вважається виконаною тільки в тому випадку, якщо вона виконується для всіх значень в результуючому стовпці підзапиту. Якщо запису підзапиту передує ключове слово ANY, то умова порівняння вважатиметься виконаною, якщо вона виконується хоча б для одного із значень в результуючому стовпці підзапиту. Якщо в результаті виконання підзапиту буде отримано порожнє значення, то для ключового слова ALL умова порівняння вважатиметься виконаною, а для ключового слова ANY — невиконаною. Згідно

стандарту ISO додатково можна використовувати ключове слово SOME, що є синонімом ключового слова ANY.

Всі приклади мають одне і те ж важливе обмеження: що стовпці, які розташовуються в результуючій таблиці завжди вибираються з однієї таблиці. Проте у багатьох випадках цього виявляється недостатньо. Для того, щоб об'єднати в результуючій таблиці стовпці з декількох початкових таблиць, необхідно виконати операцію **з'єднання.** В мові SQL операція з'єднання використовується для об'єднання інформації з двох таблиць за допомогою утворення пар зв'язаних рядків, вибраних з кожної таблиці.

Якщо необхідно отримати інформацію більш ніж з однієї таблиці, то можна або застосувати підзапит, або виконати з'єднання декількох таблиць. Якщо результуюча таблиця запиту повинна містити стовпці з різних початкових таблиць, то доцільно використовувати механізм з'єднання таблиць. Для виконання з'єднання достатньо в параметрі FROM вказати імена двох і більш таблиць, розділивши їх комами, після чого включити в запит параметр WHERE з визначенням стовпців, що використовуються для з'єднання вказаних таблиць. Крім того, замість імен таблиць можна використовувати призначені їм в параметрі FROM **псевдоніми** (аліаси).

Ключові слова EXISTS і NOT EXISTS призначені для використовування тільки спільно з підзапитами. Результат їх обробки є логічним значенням TRUE або FALSE. Для ключового слова EXISTS результат рівний TRUE в тому і лише в тому випадку, якщо в результуючій таблиці, що повертається під запитом, присутній хоча б один рядок. Якщо результуюча таблиця підзапиту порожня, результатом обробки ключового слова EXISTS буде значення FALSE. Для ключового слова NOT EXISTS використовуються правила обробки, зворотні по відношенню до ключового слова EXISTS.

Оскільки за ключовими словами EXISTS і NOT EXISTS перевіряється лише наявність рядків в результуючій таблиці підзапиту, то ця таблиця може містити довільну кількість стовпців.

## 5.3. Створення підсумкового запиту (Total Query) з використанням статистичних (узагальнюючих) функцій

Основною особливістю підсумкового запиту – є групові операції. На зразку вказано поле “Групповая операция” (рис. 5.1), але за замовчуванням це поле у конструкторі запитів не відображається. Для відображення цього поля на панелі інструментів необхідно натиснути кнопку  .

**Приклад 5.1**. Створити підсумковий запит. Для цьго потрібно створити новий запит у режимі конструктора вибравши в ньому необхідні поля (Типи_обладнання, Приєднання, Установка_ лічильників) з таблиць **Лічильники і Типи обладнання**. як зазначено на зразку (рис. 5.1). Після чого у режимі конструктора запиту клацнути на кнопку *Групповые операции* (∑ ) (Totals). У бланку запиту з'являється новий рядок *Групповая операция*. Заповніть поля цього рядка так, як вказано у прикладі (рис. 5.1), застосувавши до значень поля Установка_лічильників узагальнюючу функцію Count.

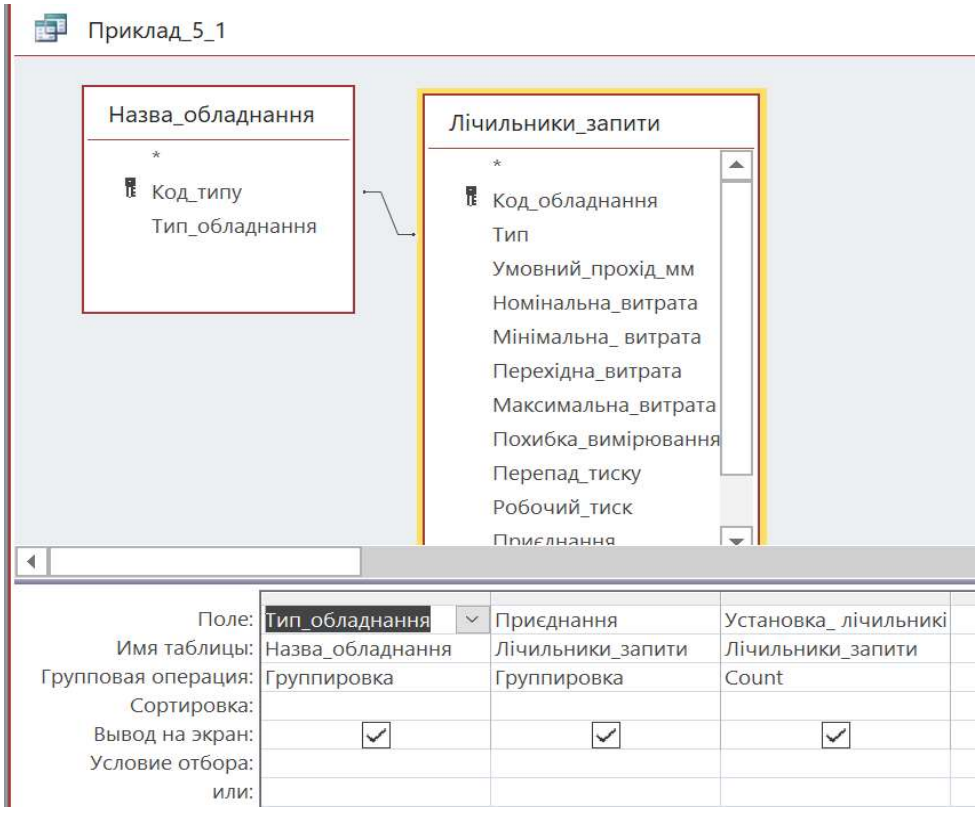



Рис. 5.1 Вигляд запиту для приклада 5.1 в режимі конструктора

1. Запустити запит на виконання та переглянути результати запиту і проаналізувати їх.

2. Переглянути та проаналізувати SQL – текст для створеного запиту.

Запит прикладу 5.1 в режимі SQL має наступний вигляд :

***SELECT Назва_обладнання.Тип_обладнання,***
***Лічильники_запити.Приєднання, Count(Лічильники_запити.[Установка_ лічильників]) AS [Count-Установка_ лічильників]***
***FROM Назва_обладнання INNER JOIN Лічильники_запити ON***
***Назва_обладнання.Код_типу = Лічильники_запити.Тип***
***GROUP BY Назва_обладнання.Тип_обладнання,***
***Лічильники_запити.Приєднання;***

Результат виконання підсумкового запиту повинен мати наступний вигляд (рис. 5.2):

Приклад_5_1

| Тип обладнання | Приєднання | Count-Устан |
|---|---|---|
| Лічильник багатоструменевий крильчатий | різьбове | 4 |
| Лічильник багатоструменевий крильчатий | фланцеве | 2 |
| Лічильник ВЄПС | різьбове | 3 |
| Лічильник ВЄПС | фланцеве | 5 |
| Лічильник одноструменевий крильчатий | різьбове | 6 |
| Ультразвуковий лічильник | фланцеве | 5 |

Рис. 5.2. Віртуальна таблиця з результатом виконання запиту для приклада 5.1

**Приклад 5.2**. Створити підсумковий запит у режимі конструктора вибравши в ньому необхідні поля (Тип_обладнання, На_складі) з таблиць **Обладнання і Типи_обладнання**. Як зазначено на зразку (рис. 5.3), застосувавши до значення поля На_складі узагальнюючу функцію Sum. Переглянути та проаналізувати SQL – текст для створеного запиту. Порівняйте результати виконання двох підсумкових запитів.

Запит прикладу 5.2 в режимі SQL має наступний вигляд:

***SELECT Назва_обладнання.Тип_обладнання,***
***Sum(Склад_обладнання.Кількість_на_складі) AS [Sum-Кількість_на_складі]***
***FROM Назва_обладнання INNER JOIN Склад_обладнання ON***
***Назва_обладнання.Код_типу = Склад_обладнання.Код_типу***
***GROUP BY Назва_обладнання.Тип_обладнання;***

Рис. 5.3. Вигляд запиту для приклада 5.2 в режимі конструктора

Результат виконання підсумкового запиту повинен мати наступний вигляд (рис. 5.4):

Приклад_5_2

| Тип обладнання | Sum-Кількість_на_складі |
|---|---|
| Балансировочний клапан | 646 |
| Запобіжний клапан | 1983 |
| Заслінка Баттерфляй поворотна дискова BORATI | 293 |
| Заслінка Баттерфляй поворотна нж | 704 |
| Заслінка поворотна диск нж підпружинена | 371 |
| Заслінка поворотна диск чавун | 322 |
| Зворотний клапан з кулькою типу 50 | 784 |
| Зворотний клапан латунний з внутрішньою різьбою типу 601 | 338 |
| Зворотний клапан тарільчатий типу 802 | 625 |
| Зворотний клапан тарільчатий типу 895 | 376 |
| Зворотний клапан чавунний типу 402 | 610 |
| Кульовий кран бр. повно прохідний з вн. різьбою типу X3222 | 347 |
| Кульовий кран з внутрішньою різьбою | 448 |
| Кульовий кран з нержавіючої сталі з вн. різьбою типу X1666 | 387 |
| Кульовий кран повно прохідний з внутрішньою різьбою типу X2777 | 492 |
| Кульовий кран типу MINI | 157 |
| Лічильник багатоструменевий крильчатий | 355 |
| Лічильник ВЄПС | 419 |
| Лічильник одноструменевий крильчатий | 298 |
| Редукційний клапан | 546 |

Рис. 5.4. Віртуальна таблиця з результатом виконання запиту для приклада 5.2

## 5.4. Перехресний запит (Crosstab Query)

Перехресні запити – це запити, в яких виконується статистична обробка даних, результати якої виводяться у вигляді таблиці. Результат виконання запиту нагадує зведену таблицю .

Перехресний запит повинен включати хоча б наступні три параметри: заголовки рядків, заголовки стовпців і поля значень. У якості даних для створення перехресного запиту може бути використано будь-який попередньо створений запит на вибірку. До поля значень якого і застосовуються статистичні функції ACCESS.

**Приклад 5.3.** Визначити мінімальну ціну клапанів кожної фірми виробника, які зберігаються. на складі (відношення Склад_обладнання). Для розв’язку цієї задачі має бути створений перехресний запит.

В структурі перехресного запиту рядками результуючої таблиці мають бути значення поля Тип обладнання , а стовпчиками – значення поля Виробник.

На перетині рядка і стовпчика і буде знаходитися обчислена за допомогою стандартної функції Min - мінімальна ціна клапанів кожної фірми-виробника.

Для створення перехресного запиту спочатку необхідно створити допоміжний запит. Створимо запит **Допоміжний_запит_5_3,** відповідно до зразка який представлено на рис. 5.5.

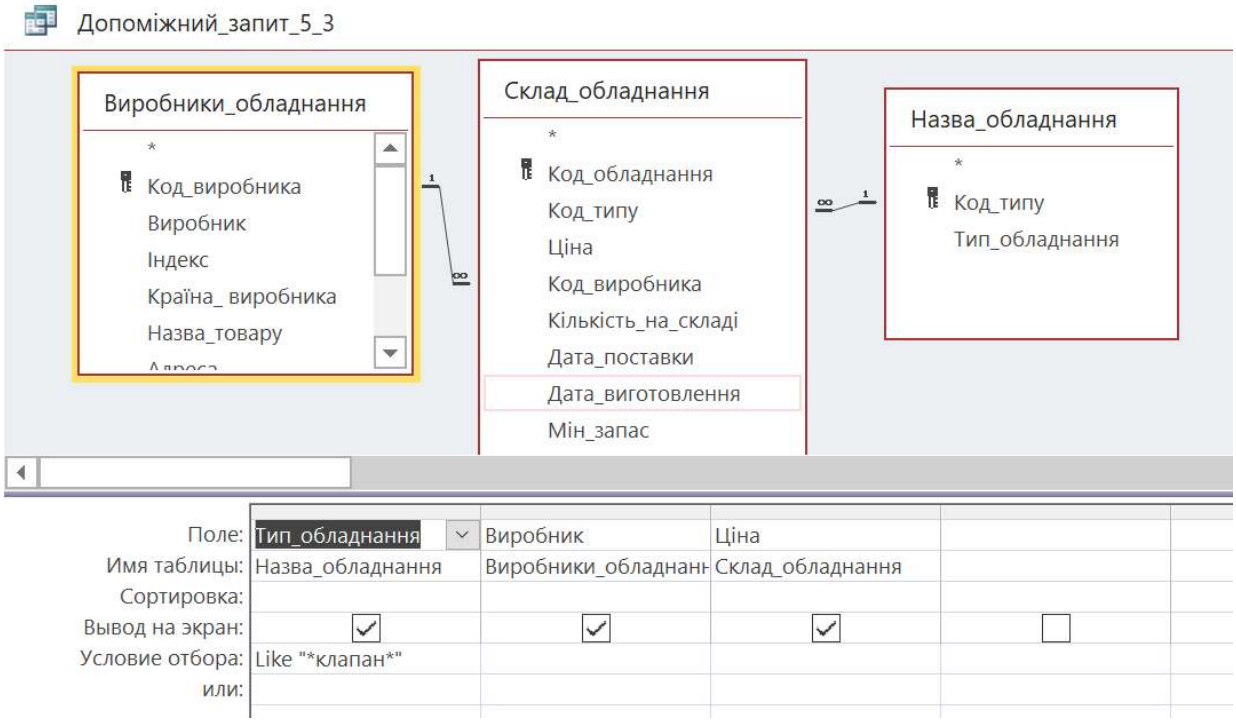



Рис. 5.5. Вигляд в режимі конструктора допоміжного запиту приклада 5.3, який буде джерелом даних для перехресного запиту

Після запуску створеного допоміжного запиту на виконання , необхідно виконати наступні пункти.

1. Вибрати в меню вікна бази даних **Создание/ Конструктор запросов,** створити звичайний запит на вибірку і потім натиснути пікторaму **Перекрестный** (рис. 5.6).


Рис. 5.6. Вікно запуску конструктора для створення перехресного запиту

2.У якості джерела даних вибрати запит **Допоміжний_запит_5_3**. Перетягнути потрібні поля з таблиць у поля запиту та встановити:

- для поля **Тип_обладнання** групову операцію **“Группировка”** та перехресну таблицю **“Заголовки строк”**;

- для поля **Виробник** групову операцію **“Группировка”**, перехресну таблицю **“Заголовки столбцов”**;

- для наступного поля групову операцію **Min** та у перехресну таблицю **“Значение”** (рис. 5.7);

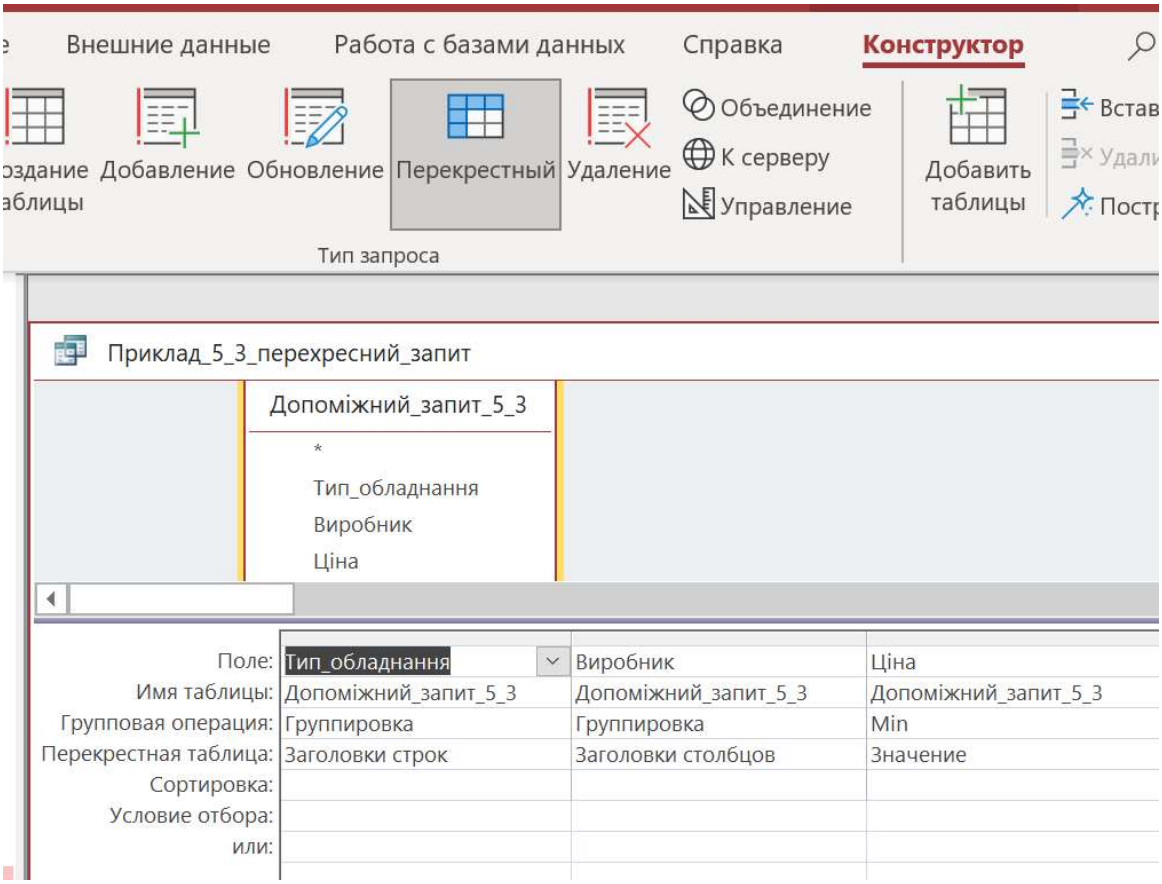



Рис. 5.7. Вигляд в режимі конструктора створення перехресного запиту

Запит у вигляді SQL- рядка має наступний вигляд:

***TRANSFORM Min(Допоміжний_запит_5_3.Ціна) AS [Min-Ціна]***

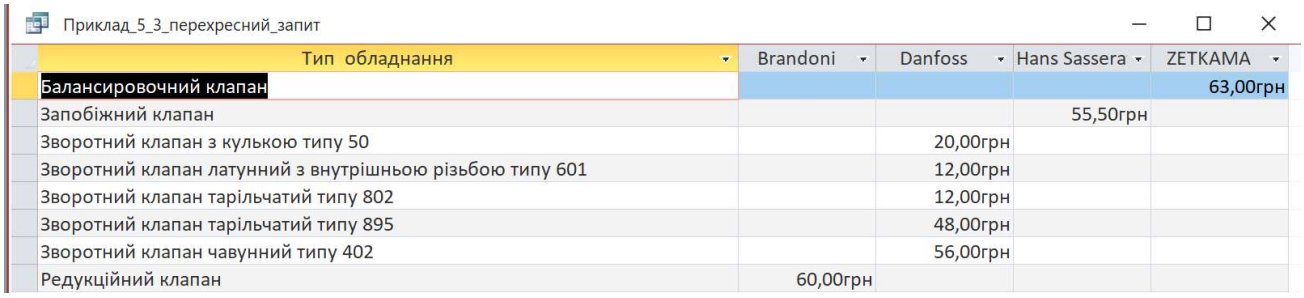
***SELECT Допоміжний_запит_5_3.Тип_обладнання***

***FROM Допоміжний_запит_5_3***

***GROUP BY Допоміжний_запит_5_3.Тип_обладнання***

***PIVOT Допоміжний_запит_5_3.Виробник;***

Результат виконання перехресного запиту прикладу 5.3 повинен мати наступний вигляд (рис.5.8):



Приклад_5_3_перехресний_запит

| Тип обладнання | Brandoni | Danfoss | Hans Sassera | ZETKAMA |
|---|---|---|---|---|
| Балансировочний клапан | | | | 63,00грн |
| Запобіжний клапан | | | 55,50грн | |
| Зворотний клапан з кулькою типу 50 | | 20,00грн | | |
| Зворотний клапан латунний з внутрішньою різьбою типу 601 | | 12,00грн | | |
| Зворотний клапан тарільчатий типу 802 | | 12,00грн | | |
| Зворотний клапан тарільчатий типу 895 | | 48,00грн | | |
| Зворотний клапан чавунний типу 402 | | 56,00грн | | |
| Редукційний клапан | 60,00грн | | | |

Рис. 5.8. Результати виконання перехресного запиту

## 5.5. Створення параметричного запиту

За допомогою параметричних запитів можна автоматизувати процес зміни критеріїв у часто використовуваних запитах. Параметричний запит при кожному виконанні потребує введення значень параметра, який був визначеним при конструюванні відповідного запита. Це виключає потребу постійної модифікації запиту у режимі конструктора для зміни умови критерію пошуку. При побудові таких запитів не існує обмежень на кількість параметрів, тобто можна створити запит, що вимагатиме введення декількох параметрів за якими і буде проводися пошук необхідної інформації.

Параметричні запити зручно використовувати у формах та звітах, оскільки при відкритті відповідного об’єкту Access вимагає від користувача введення заданого у запиті параметра.

**Приклад 5.4.** Створити параметричний запит , який буде виводити назву фірми виробника, назву обладнання та ціну одиниці обладнання. У якості параметра вибрати поле Виробник .

1. Для створення параметричного запиту потрібно вибрати в меню вікна бази даних **Создание/ Конструктор** запросов, створити звичайний запит на вибірку і потім натиснути пікторaму **Параметры** (рис. 5.9).


Рис. 5.9. Вибір опції меню Параметры для створення параметричного запиту

2. У якості джерела даних виберіть таблиці Типи_обладнання, Виробник обладнання та Обладнання.
3. Перетягніть потрібні поля з таблиць у поля запиту та встановіть для поля **Виробник** умову відбора **[Виробник]** (рис. 5.10).

Після вибору опції меню Параметри з’явиться вікно наведене нижче, у якому потрібно вибрати параметр та його тип. Таких параметрів можна вибрати декілька.

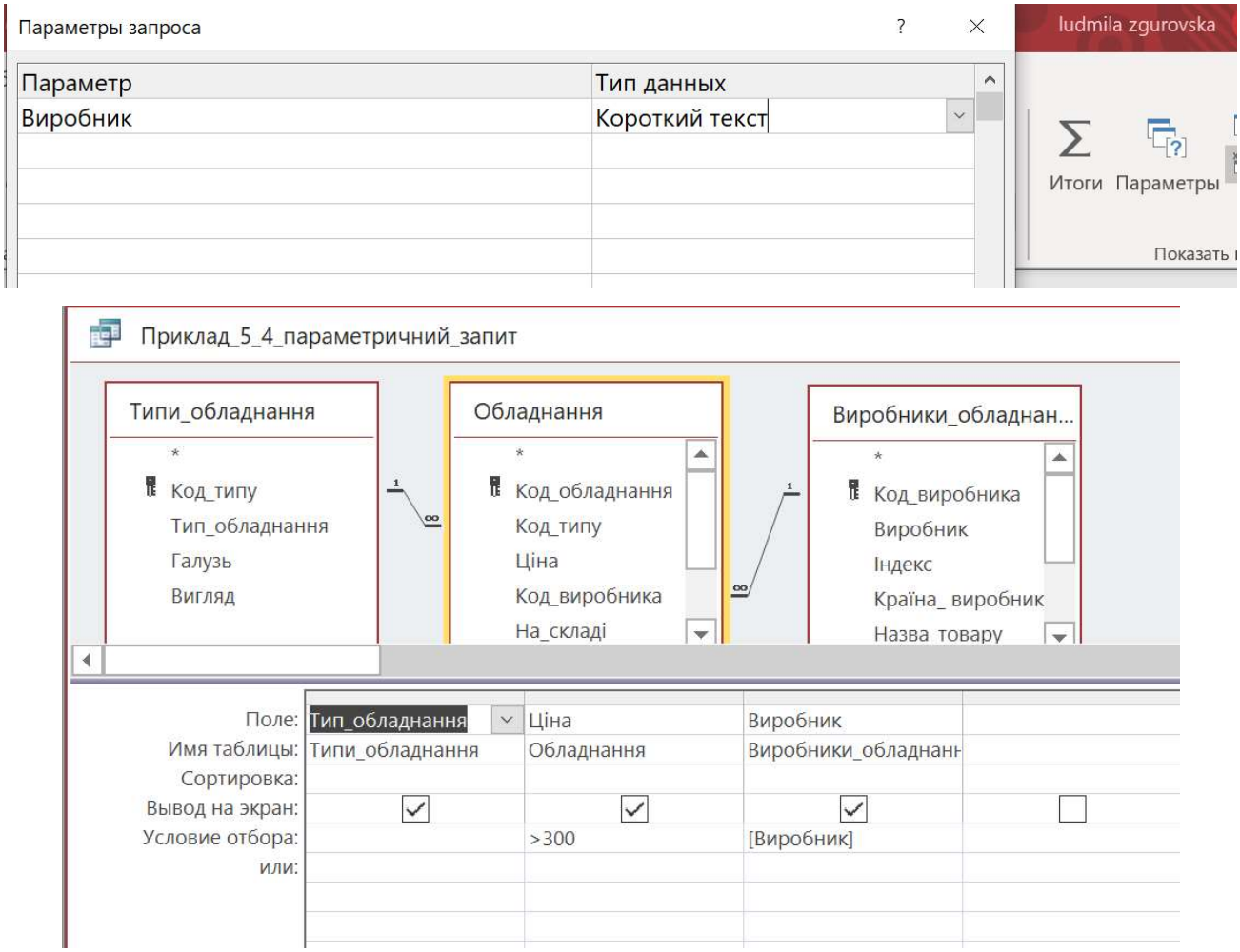



Рис. 5.10. Вигляд в режимі конструктора створення параметричного запиту

5. Після запуску запиту на виконання з’явиться діалогове вікно запиту, в якому потрібно задати назву фірми виробника, інформація про яку нас цікавить (рис. 5.11).

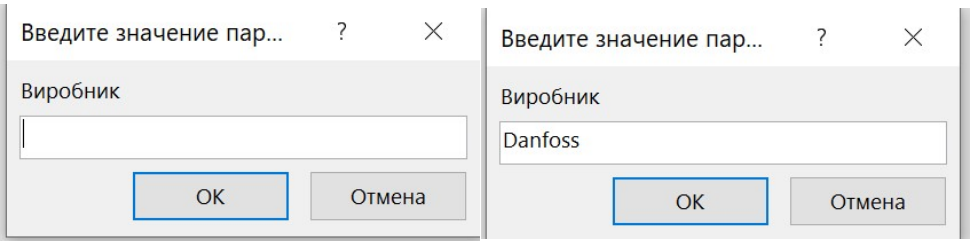



Рис. 5.11. Діалогове вікно параметричного запиту для вводу значення параметра пошуку

Результат виконання параметричного запиту повинен мати наступний вигляд (рис. 5.12):

Приклад_5_4_параметричний_запит

| Тип обладнання | Ціна | Виробник |
|---|---|---|
| Зворотний клапан чавунний типу 402 | 302,00 грн. | Danfoss |
| Зворотний клапан чавунний типу 402 | 356,00 грн. | Danfoss |
| Зворотний клапан чавунний типу 402 | 400,00 грн. | Danfoss |
| Зворотний клапан чавунний типу 402 | 426,00 грн. | Danfoss |
| Зворотний клапан чавунний типу 402 | 550,00 грн. | Danfoss |
| Зворотний клапан з кулькою типу 50 | 322,00 грн. | Danfoss |
| Фільтр сітчастий чавунний типу Y333 | 380,00 грн. | Danfoss |
| Фільтр сітчастий чавунний типу Y333 | 500,00 грн. | Danfoss |
| Фільтр сітчастий чавунний типу Y333P | 350,00 грн. | Danfoss |
| Фільтр сітчастий чавунний типу Y333P | 420,00 грн. | Danfoss |
| Фільтр сітчастий чавунний типу Y333P | 520,00 грн. | Danfoss |

Рис. 5.12. Результат виконання параметричного запиту

Перегляньте та проаналізуйте SQL – текст для створеного запиту. Запит у вигляді SQL- рядка повинен мати наступний вигляд:

***PARAMETERS Виробник Text ( 255 );***
***SELECT Виробники_обладнання.Виробник,***
***Типи_обладнання.Тип_обладнання, Обладнання_для_лек_6_дати.Ціна***
***FROM (Обладнання_для_лек_6_дати INNER JOIN Виробники_обладнання***
***ON Обладнання_для_лек_6_дати.Код_виробника =***
***Виробники_обладнання.Код_виробника) INNER JOIN Типи_обладнання ON***
***Обладнання_для_лек_6_дати.Код_типу = Типи_обладнання.Код_типу***
***WHERE (((Виробники_обладнання.Виробник)=[Виробник]) AND***
***((Обладнання_для_лек_6_дати.Ціна)>300));***

## 5.6. Прийняття рішень за допомогою функції IIF

Функція має наступний формат

**IIF (Умова, Значення 1, Значення 2)**

Умова – логічний вираз, що повертає значення TRUE або FALSE ;

Значення 1 - значення, що повертається функцією за умови прийняття логічним виразом значення TRUE;

Значення 2 – значення, що повертається функцією за умови прийняття логічним виразом значення FALSE .

**Приклад 5.5**. Створити запит для розв'язання наступного завдання: якщо залишок виробів на складі менший за мінімальний запас, то виводити повідомлення «Замовити», інакше «Не замовляти»

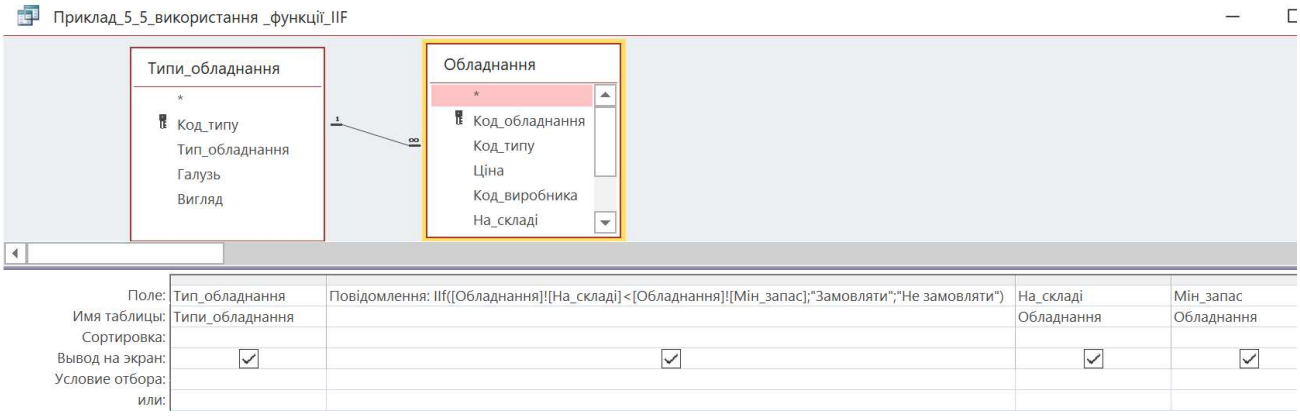



Рис. 5.13. Вигляд в режимі конструктора створення запиту на вибірку з використанням функції IIF

Для запису функції **IIF** використайте **Построитель выражений** (рис. 5.14).

Запит у вигляді SQL- рядка для приклада 5.5 повинен мати наступний вигляд

***SELECT Типи_обладнання.Тип_обладнання,***
***IIf([Обладнання]![На_складі]<[Обладнання]![Мін_запас],"Замовляти","Не замовляти") AS Повідомлення, Обладнання.На_складі,***
***Обладнання.Мін_запас***
***FROM Типи_обладнання INNER JOIN Обладнання ON***
***Типи_обладнання.Код_типу = Обладнання.Код_типу;***

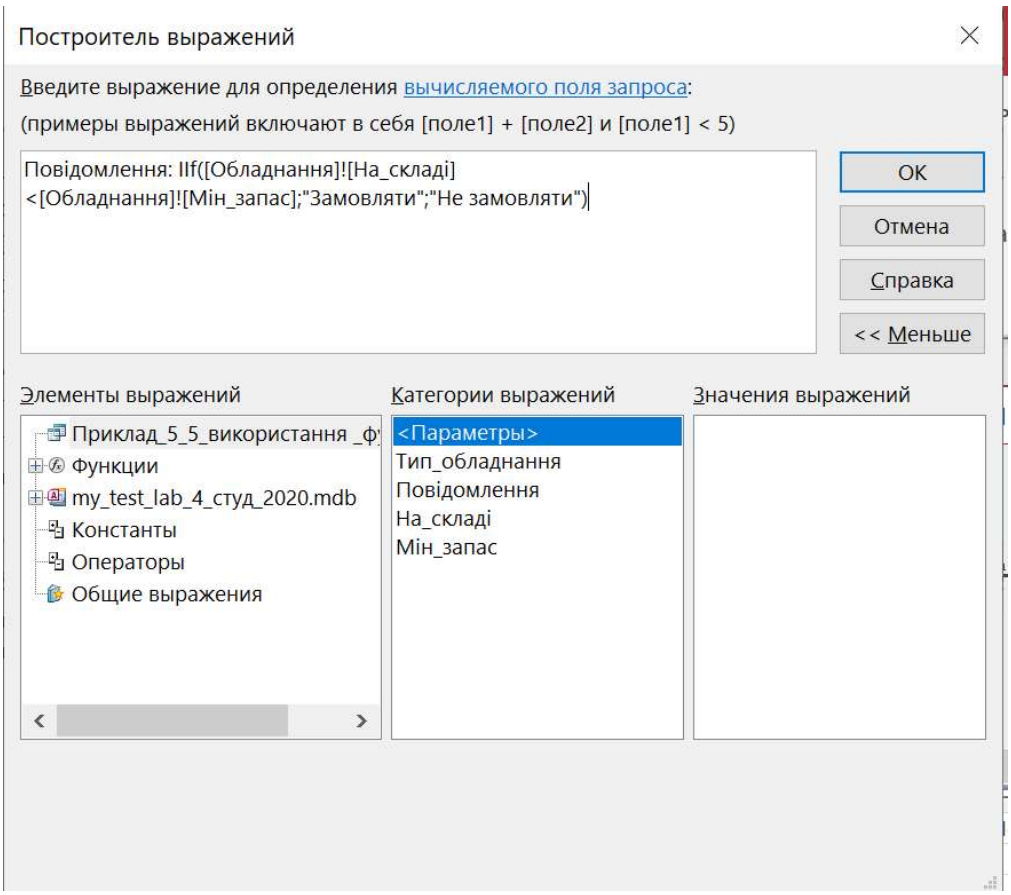



Рис. 5.14. Створення виразу з використанням функції IIF в **Построителе выражений**

Приклад_5_5_використання _функції_IIF

| Тип обладнання | Повідомлен | На_складі | Мін_запас |
|---|---|---|---|
| Запобіжний клапан | Замовляти | 4 | 12 |
| Кульовий кран з внутрішньою різьбою | Замовляти | 4 | 7 |
| Зворотний клапан тарільчатий типу 895 | Замовляти | 5 | 15 |
| Запобіжний клапан | Замовляти | 6 | 10 |
| Запобіжний клапан | Замовляти | 6 | 17 |
| Зворотний клапан латунний з внутрішньою різьбою типу 601 | Не замовляти | 7 | 7 |
| Запобіжний клапан | Замовляти | 7 | 16 |
| Лічильник одноструменевий крильчатий | Замовляти | 8 | 18 |
| Фільтр сітчастий чавунний типу Y333 | Не замовляти | 9 | 9 |
| Запобіжний клапан | Не замовляти | 11 | 11 |
| Запобіжний клапан | Не замовляти | 11 | 11 |
| Кульовий кран бр. повно прохідний з вн. різьбою типу X3222 | Не замовляти | 12 | 10 |
| Ультразвуковий лічильник | Не замовляти | 12 | 10 |
| Зворотний клапан з кулькою типу 50 | Не замовляти | 12 | 11 |
| Запобіжний клапан | Не замовляти | 12 | 12 |
| Кульовий кран з нержавіючої сталі з вн. різьбою типу X1666 | Не замовляти | 12 | 9 |
| Лічильник ВЄПС | Замовляти | 12 | 16 |

Рис. 5.15. Результат виконання запиту з використанням функції IIF

**Приклад 5.6**. Створити запит для розв’язання наступного завдання: якщо кількість обладнання на складі дорівнює 11, то змінити дату у полі Очікується

на 7 днів більшу за поточну дату, інакше дату у полі Очікується залишити незмінною. Запит у режимі конструктора наведено на рис. 5.16.

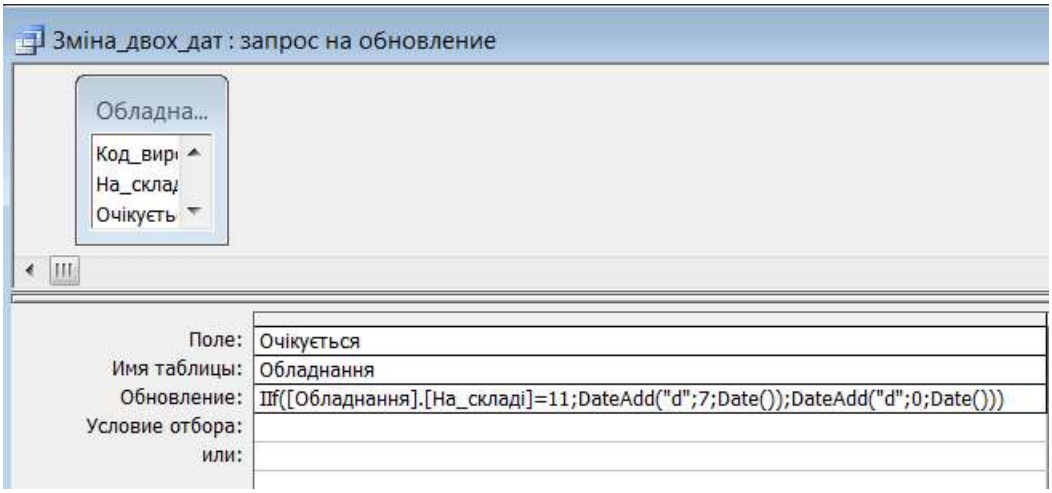



Рис. 5.16. Функція IIf, аргументами якої є функції Дата/время

Запит у вигляді SQL- рядка для приклада 5.6.

***UPDATE Обладнання SET Обладнання.Очікується = IIf([Обладнання].[На_складі]=11,DateAdd("d",7,Date()),DateAdd("d",0,Date())) ;***

Функція Date() повертає поточну дату у короткому форматі, а функція DateAdd () додає значення 7 до відповідного параметра поточної дати ( в нашому прикладі – це день).

## 5.7. Функції для обробки даних типу Дата/время

Дані типу **Дата/время** можуть приймати значення для визначення років в інтервалі від 100 до 9999 та займають в пам’яті об’єм 8 байт. Нагадаю, що сталі типу **Дата/время** мають починатися і закінчуватися символом решітка, наприклад, #15/01/2021#.

Для роботи з такими даними в ACCESS існує велика кількість вбудованих спеціальних функцій. Розглянемо деякі з них:

Таблиця 5.2. Функції дати

| Функція | Призначення функції |
|---|---|
| Format() | можна використовувати для вибору формата виведення дати |
| Date() | повертає поточну дату у короткому форматі. |
| Now() | повертає поточні дату і час |
| DateAdd(); | використовується для додавання вказаного значення до відповідного параметра дати або його зменшення на вказане число. |
| Day() | повертає число місяця |
| WeekDay() | повертає числове значення дня тижня |
| Month() | повертає порядковий номер місяця |
| Year() | повертає рік |
| DatePart(0) | функція може бути використана для отримання частки конкретної дати, наприклад, дня, місяця або року |
| DateDiff() | функція може бути використана для отримання різниці або інтервалу часу між двома датами ( в днях, місяцях, роках або у таких одиницях часу як секунди, хвилини, години) |
| DateValue(0 | перевіряє чи введений рядок є допустимою датою |
| DateSerial() | повертає значення дати введених параметрів року, місяця і дня |

## 5.8. Приклади використання функцій Дата/время

**Приклад 5.7.** Створити запит, що повертає поточну дату у довгому форматі (5 березня 2021 року).

***SELECT Format (Now(), "dd mmmm yyyy") AS***
***Приклад_використання_Format;***

Результат виконання запиту представлено на рис. 5.17

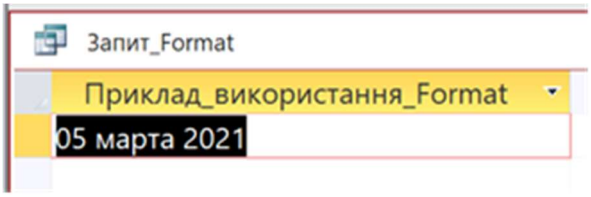



Рис. 5.17. Результат виконання запиту з використанням функції Format

Функція **DatePart()** має додатковий параметр, який дозволяє відобразити необхідну частину дати. В прикладі використано значення параметра "q", який відображає номер квартала. Можна використовувати і інші параметри, наприклад, рік - "yyyy", квартал - "q", "m", номер місяця, день - "d", тиждень - "w", годину - "h", хвилини - "n", секунди - "s" тощо).

**Приклад 5.8.** Визначити в якому кварталі було виготовлено обладнання. Для реалізації доцільно використати функцію **DatePart()**

***SELECT Лічильники_дата_виготовлення.Дата_виготовлення, DatePart("q", Лічильники_дата_виготовлення.Дата_виготовлення) AS Квартал***
***FROM Лічильники_дата_виготовлення;***

Результат цього запиту має вигляд.

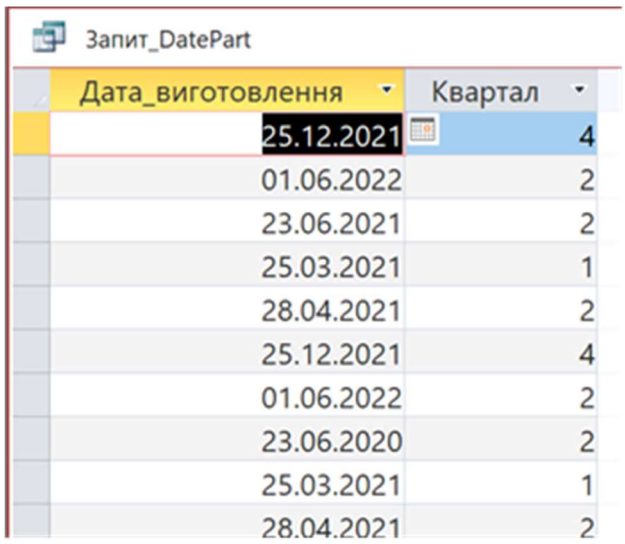



Рис. 5.18. Результат запиту з використанням функції DatePart() для визначення кварталу

**Приклад 5.9.** Аргументом функцій Day, Weekday, Month, Year є змінна типу Дата/время. В табл. 5.1 наведено призначення цих функцій. Застосуємо функції до поля Очікється та переглянемо результат виконання запиту, представленого в режимі конструктора на рис. 5.19.

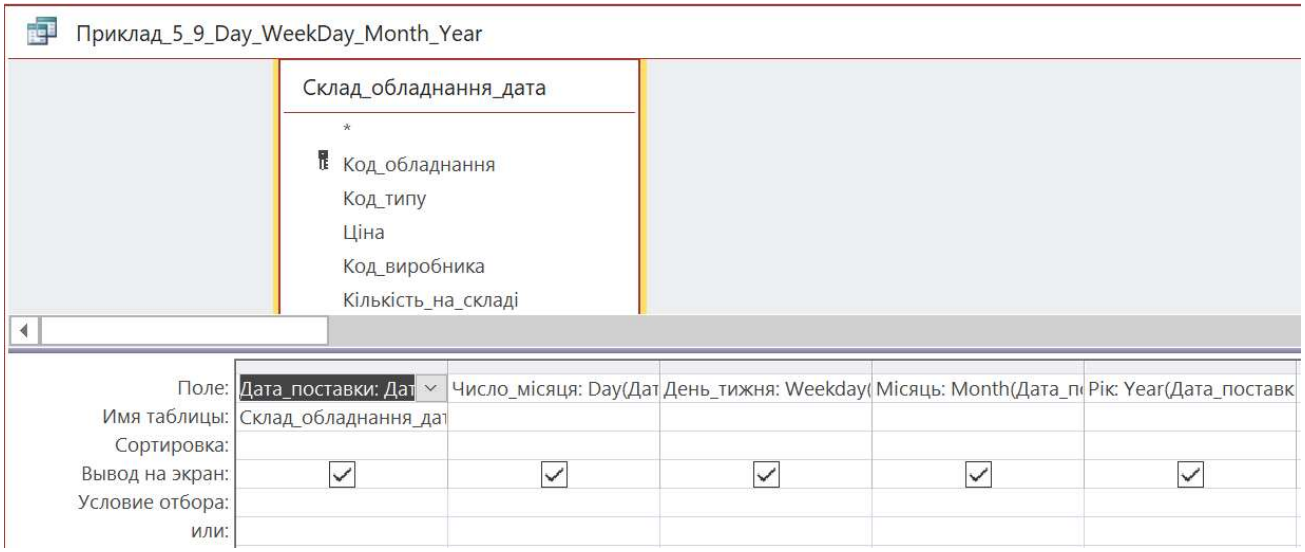



Рис. 5.19. Вигляд запиту для приклада 5.9 в режимі конструктора

Запит у вигляді SQL- рядка має наступний вигляд:

***SELECT Склад_обладнання_дата.Дата_поставки AS Дата_поставки, Day(Дата_поставки) AS Число місяця, Weekday(Дата_поставки) AS День_тижня, Month(Дата_поставки) AS Місяць, Year(Дата_поставки) AS Рік***
***FROM Склад_обладнання_дата;***

Результат виконання запиту має наступний вигляд (наведена частина даних).

Приклад_5_9_Day_WeekDay_Month_Year

| Дата_постав | Число_місяц | День_тижня | Місяць | Рік |
|---|---|---|---|---|
| 15.01.2022 | 15 | 7 | 1 | 2022 |
| 10.07.2022 | 10 | 1 | 7 | 2022 |
| 12.07.2022 | 12 | 3 | 7 | 2022 |
| 15.04.2022 | 15 | 6 | 4 | 2022 |
| 10.09.2022 | 10 | 7 | 9 | 2022 |
| 10.09.2022 | 10 | 7 | 9 | 2022 |
| 10.09.2022 | 10 | 7 | 9 | 2022 |
| 15.04.2022 | 15 | 6 | 4 | 2022 |
| 30.07.2022 | 30 | 7 | 7 | 2022 |
| 10.06.2022 | 10 | 6 | 6 | 2022 |
| 15.03.2022 | 15 | 3 | 3 | 2022 |
| 17.02.2022 | 17 | 5 | 2 | 2022 |

Рис. 5.20. Результат запиту з використанням функцій Day, Weekday, Month, Year

**Приклад 5.10.** Визначити дату поставки обладнання, яку збільшено на 10 днів для обладнання, мінімальний запас якого на складі дорівнює 12 одиниць. Використаємо функцію DateAdd() в команді UPDATE.

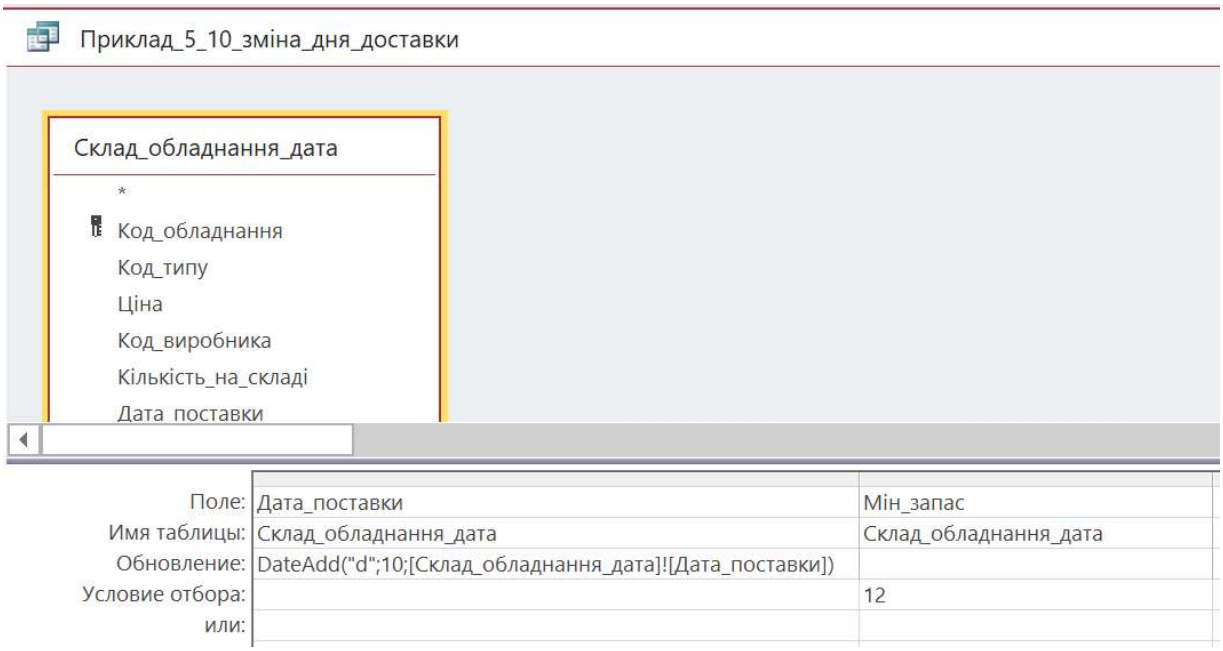



Рис. 5.21. Вигляд запиту для приклада 5.10 в режимі конструктора

***UPDATE Склад_обладнання_дата SET***
***Склад_обладнання_дата.Дата_поставки =***
***DateAdd("d",10,[Склад_обладнання_дата]![Дата_поставки])***
***WHERE (((Склад_обладнання_дата.Мін_запас)=12));***

Зверніть увагу на знаки пунктуації, використані при записі аргументів функції DateAdd в режимі конструктора і в режимі SQL. Для запису аргументів функції DateAdd в рядку Обновление в режимі конструктора було використано Построитель. При використанні Построителя, аргументи функції DateAdd відокремлюються один від одного крапкою з комою. Якщо потрібно зменшити значення року в полі Дата_поставки таблиці Склад_обладнання_дата на один рік, команда SQL матиме вигляд:

***UPDATE Склад_обладнання_дата SET***
***Склад_обладнання_дата.Дата_поставки = DateAdd("yyyy",-***
***1,[Склад_обладнання_дата]![Дата_поставки]);***

**Приклад 5.11.** Визначити скільки місяців зберігається на складі обладнання, кількість якого менша за 5 одиниць. Використаємо поля

Дата_поставки, Кількість_на_складі таблиці Склад_обладнання_дата та поле Тип_обладнання таблиці Назва_обладнання. До поля Дата_поставки застосуємо функцію **DateDiff ().**

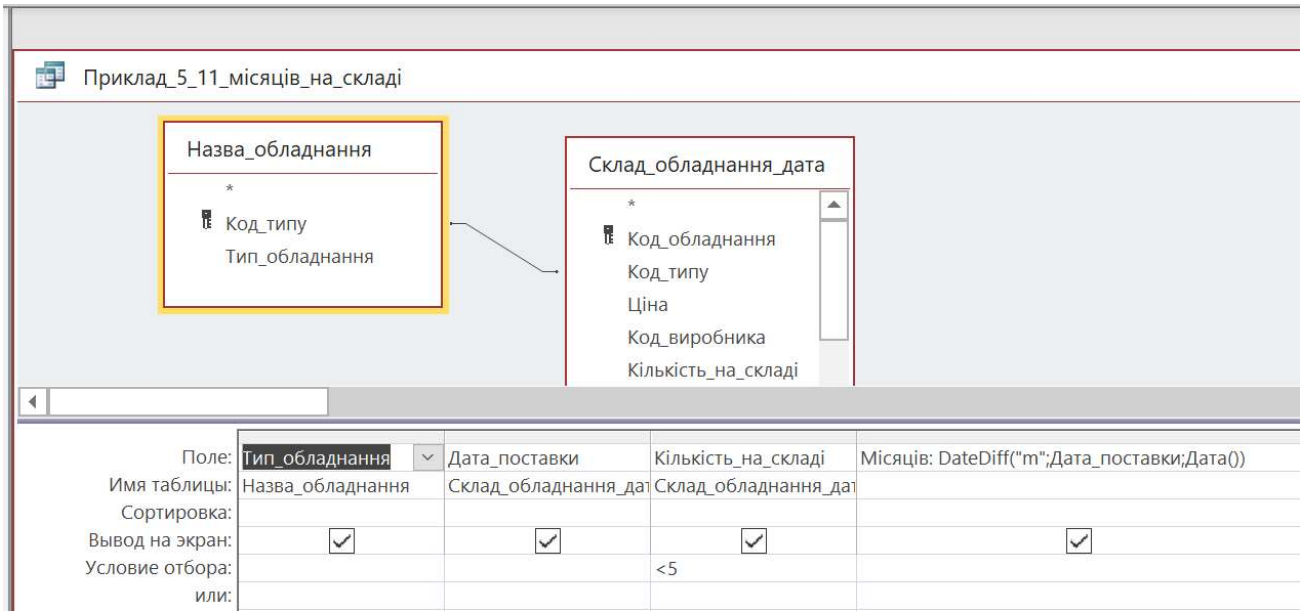
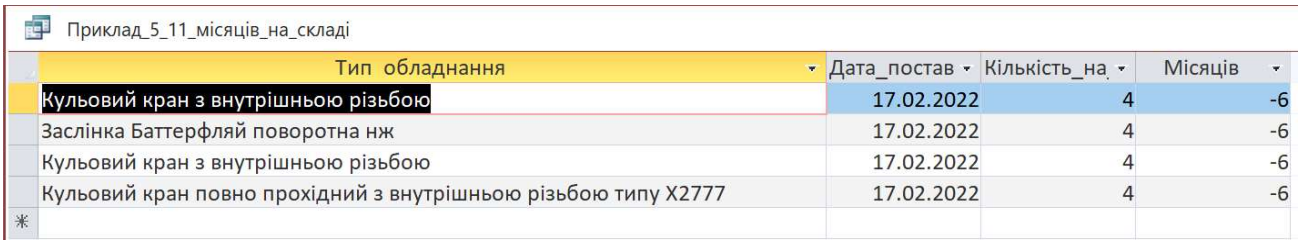



Рис. 5.22. Вигляд запиту для приклада 5.11 в режимі конструктора

Команда SQL матиме вигляд

***SELECT Назва_обладнання.Тип_обладнання,***
***Склад_обладнання_дата.Дата_поставки,***
***Склад_обладнання_дата.Кількість_на_складі, DateDiff("m",***
***Дата_поставки, Date()) AS Місяців***
***FROM Склад_обладнання_дата INNER JOIN Назва_обладнання ON***
***Склад_обладнання_дата.Код_типу = Назва_обладнання.Код_типу***
***WHERE (((Склад_обладнання_дата.Кількість_на_складі)<5));***

Цей запит повертає з таблиці Склад_обладнання_дата назву обладнання, дату поставки та значення кількості місяців, що це обладнання знаходиться на складі.




| Тип обладнання | Дата_постав | Кількість_на | Місяців |
|---|---|---|---|
| Кульовий кран з внутрішньою різьбою | 17.02.2022 | 4 | -6 |
| Заслінка Баттерфляй поворотна нж | 17.02.2022 | 4 | -6 |
| Кульовий кран з внутрішньою різьбою | 17.02.2022 | 4 | -6 |
| Кульовий кран повно прохідний з внутрішньою різьбою типу X2777 | 17.02.2022 | 4 | -6 |

Рис. 5.20. Результат запиту прикладу 5.11 з використанням функцій DateDiff()

**Приклад 5.12.** Приклад використання функції DateValue()

Функція DateValue() перевіряє, чи введений рядок є допустимою датою. Якщо введений рядок розпізнається як допустима дата, то значення дати повертається у короткому форматі.

***SELECT DateValue("28.03.2022") AS Вбудований_формат_дати;***

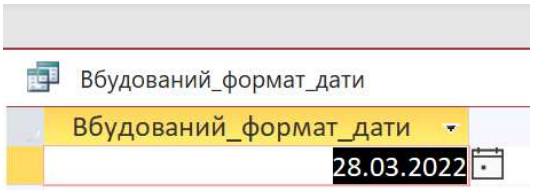



Рис. 5.23. Результат запиту прикладу 5.11 з використанням функцій DateValue ()

Вибір даних про обладнання, яке було виготовлене в останньому кварталі 2020 року

***SELECT ****
***FROM Лічильники***
***WHERE Дата_виготовлення >= DateValue("01.10.2021") AND***
***Дата_виготовлення <= DateValue("31.12.2021");***

**Приклад 5.13.** Функція DateSerial() повертає значення дати введених параметрів : року, місяця і дня.

Параметри, що вводяться можуть бути виразами, які включають арифметичні операції.

Функція DateSerial() аналізує вирази у введених параметрах перед тим, як повертає дату-результат.

***SELECT DateSerial( 2021, 04, 1-1) AS Приклад_DateSerial;***

Результат

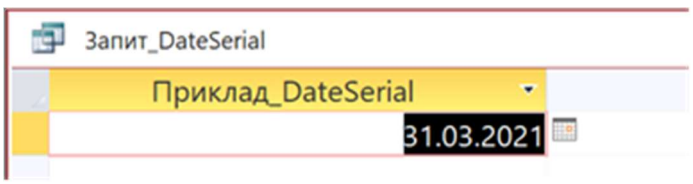



Рис. 5.24.

Функцію **DateSerial()** можна використовувати для виконання обчислень. Наприклад, якщо необхідно визначити кількість днів, на яку була відстрочена поставка обладнання можна записати вираз:

***DateSerial(Year([Дата_виконання]);Month([Дата_виконання]);Day([Дата_виконання]))-***
***DateSerial(Year([Очікувана_дата]);Month([Очікувана_дата]);Day([Очікувана_дата]))***

Набрати такий вираз можна за допомогою Построителя. Потім вставити вираз у властивість поля Данные відповідної форми. Саме поле має бути попередньо створене на формі як свободное. При відкритті форми до цього поля буде виводитися результат виразу.

**Приклад 5.14.** Якщо потрібно заповнити даними одне поле таблиці, значення яких вже зберігаються в іншій таблиці можна використати запит на оновлення такого виду.

***UPDATE Склад_обладнання_ціна_товару INNER JOIN Затвори_клапани ON Склад_обладнання_ціна_товару.Код_обладнання =***
***Затвори_клапани.Код_обладнання SET Затвори_клапани.Ціна =***
***[Склад_обладнання_ціна_товару]![Ціна]***
***WHERE***
***(((Затвори_клапани.Код_обладнання)=[Склад_обладнання_ціна_товару]![Код_обладнання]));***

В цьому прикладі поле **Ціна** таблиці **Затвори_Клапани** заповнюється даними з поля **Ціна** попередньо створеного запиту **Склад_обладнання_ціна_товару**.

## 5.9. Завдання до комп'ютерного практикуму

*Завдання частина перша таблиці*

Виконати всі приклади, наведені у теоретичних відомостях та зберегти їх з іменами, наприклад, Приклад_Format_FFF, Приклад_DateValue_FFF і т.д.(виконати приклади з усіма функціями дати), Приклад_5_1_FFF, де FFF -

прізвище студента. Записати до протоколу послідовність дій (опції меню, критерій конструктора запитів та текст SQL-команди), які необхідно виконати для отримання результату та самі результати.

Виконання операції «Сжатие базы данных»

Після завершення роботи з базою даних в Access необхідно завжди виконувати **Сжатие базы данных,** після чого об'єм бази даних суттєво зменшується. В різних версіях Access ця опція знаходиться в різних пунктах меню. Наприклад, існує така можливість

**Для сжатия файла базы данных Access необходим эксклюзивный доступ к нему.**

1. Запустите **Access**.
2. В меню Сервис выберите пункт Служебные программы, а затем — пункт **Сжать** и восстановить **базу данных**. ...
3. В диалоговом окне **База данных** для **сжатия** выберите файл, который требуется **сжать**, и нажмите кнопку **Сжать**.

Більше... • 3 трав. 2017 р.

Або наступний варіант. Обрати пункт меню Работа с базой данных, а потім Сжать или восстановить базу данных


**Завдання частина друга**

Виконати наступні завдання та зберегти їх з іменами, наприклад, Завдання_5_1_FFF. Де FFF - прізвище студента. Записати до протоколу послідовність дій (опції меню, критерій конструктора запитів та текст SQL-команди), які необхідно виконати для отримання результату та результати виконання запитів. Всі запити, що змінюють значення даних в таблицях, потрібно виконувати до копій цих таблиць. Такі таблиці потрібно попередньо створити.

1. Створити запит для виконання завдання: якщо залишок товару на складі складає менше ніж 5 одиниць за мінімально можливий запас, встановити значення поля Дата_поставки більшим на 7 днів за поточну дату.

2. Створити запит для виконання завдання: збільшити кількість товарів **(**відношення Склад_обладнання_додано**)**, що зберігаються на складі (Поле Кількість_на_складі) на 40, якщо залишок цього товару на складі складає менше ніж 5 одиниць за мінімально можливий запас. Таблиця результатів повинна складатися з полів Кількість_на_складі, Мін_запас.

3. Створити перехресний запит, результатом якого має бути загальна кількість лічильників для кожної фірми виробника, які зберігаються. на складі (відношення Склад_обладнання) та мають наступні параметри: фланцеве приєднання та значення максимальної витрати, що дорівнює 20 або 32. Рядками результуючої таблиці мають бути значення поля Виробник , а стовпчиками – значення поля Тип обладнання.

4. Створити параметричний запит (в якості параметрів вибрати поля умовний прохід та приєднання). Таблиця результатів повинна складатися з полів Тип обладнання, Умовний прохід, Приєднання, Ціна.

5. Створити запит для виконання завдання: обчислити скільки балансировочних клапанів фірми ZETKAMA зберігаються на складі.

6. Сформулювати завдання для створення запиту з використанням функції Avg. Створити запит та виконати його.

7. Визначити на яку суму зберігаються на складі **(**відношення Склад_обладнання**)**, клапани фірми Danfoss. Таблиця результатів повинна складатися з одного поля ЗагальнаСума.

8. Сформулювати завдання для створення запиту з використанням функції count. Створити запит та виконати його.

9. Сформулювати завдання для створення запиту з використанням логічного оператора **in.** Створити запит та виконати його.

10. Дати письмові відповіді на контрольні запитання та навести приклади.

## 5.10. Контрольні запитання

1. Поясніть своїми словами, що ви розумієте під словом підзапит?
2. Для чого використовується зовнішня і внутрішня команда SELECT?
3. Для чого використовується параметр INNER JOIN?
4. Для чого використовується параметр TRANSFORM?
5. Чи повинен бути укладений в дужки текст підзапиту?
6. Які правила і обмеження застосовуються до підзапитів?
7. Для чого використовуються з підзапитами ключові слова ANY і ALL? Створіть запит з використанням ключових слів ANY і ALL.
8. Який є результат їхньої обробки?
9. Який є результат для ключового слова EXISTS? Створіть запит з використанням ключового слова EXISTS .

# Комп'ютерний практикум №6. Створення однотабличних форм бази даних

*Мета роботи* - набути навичок створення форм бази даних за допомогою конструктора і майстра СУБД Access.

***Теоретичні відомості***

Форма Access – це об’єкт бази даних, за допомогою якого можна створити інтерфейс користувача для програми бази даних. В Access можна створювати багато варіантів форм. Для безпосередньої обробки даних створюють так звані "зв’язані" форми. "Зв’язана" форма – це форма, підключена до джерела даних, таких як таблиця чи запит, за допомогою якої можна вводити, редагувати або відображати дані з цього джерела даних. Також можна створити "вільну" форму, яка не веде прямо до джерела даних, але яка все одно містить кнопки, підписи або інші елементи керування, необхідні для роботи з програмою [3].

Структура форми (рис. 6.1) має три області **Область даних (Detail**), **Заголовок формы (Form Header), Примечание формы (Form Footer).** Останні дві області можуть бути утворені за командою контекстного меню **Заголовок/примечание формы**.

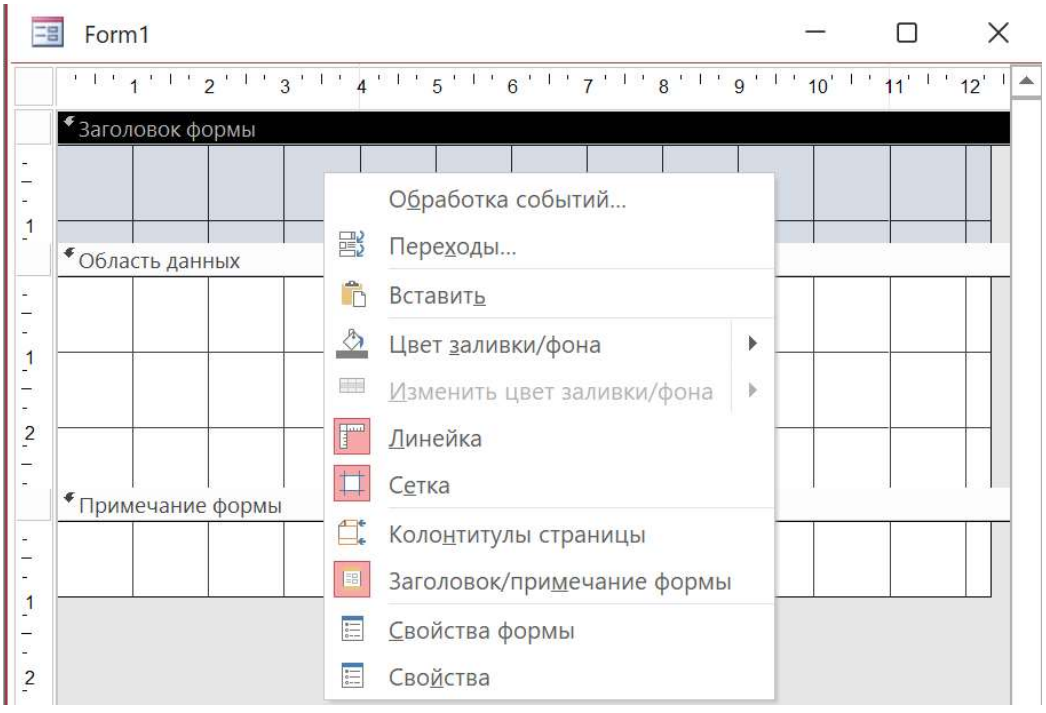



Рис. 6.1. Вигляд нової форми в режимі **Конструктор**

Області форми наповнюються різними графічними елементами. Графічні об'єкти, пов'язані з записами таблиць і призначені для відображення даних деякого поля, називаються елементами керування. Основними типами елементів керування є - Поле (Text Box), Поле зі списком (List Box), Список (Combo Box).

Графічні об'єкти, не пов'язані з таблицями або запитами, призначені насамперед для створення макета форми і містять написи полів, вбудовані об'єкти, їх написи, заголовки.

Як форма в цілому, так і кожен з її елементів має свої властивості. Властивості елемента дозволяють визначити його зовнішній вигляд, розмір, місце розташування в формі, режим введення / виведення, прив'язати до елементу вираз, макрос або програму. Модифікувати вигляд і можливості форми можна змінюючи її властивості. Форма СУБД Access має 81 визначену властивість. Доступ до властивостей будь-якого об’єкта Access, можна отримати обравши опцію **Свойства** контекстного меню або обрати піктограму **Свойства** головного меню. Контекстне меню відкривається натисканням правої кнопки миші.

У формі також можуть бути передбачені кнопки управління для різних цілей, наприклад:

• переходу до інших записів в режимі перегляду даних таблиці;

• роботи з записами (додати, оновити, дублювати, зберегти, видалити);

• роботи з формою (відкрити, закрити, фільтрувати, оновити).

Вибираючи необхідні значення властивостей форми та її елементів можна створити унікальну форму. Форми СУБД Access тісно пов'язані з даними, тому створивши форму, можна використовувати її для введення, редагування та перегляду даних відповідного джерела даних.

При роботі з формою може виконуватися обробка подій, ініційованих користувачем або наступаючих в процесі роботи з формою. Форма може створюватися для управління додатком користувача. Типові процедури формуються автоматично при створенні елементів форми. Такими елементами, наприклад, є, графічні кнопки, з якими можуть зв'язуватися події наступних категорій:

- перехід по записах, обробка записів (додавання, видалення, друк, відновлення);
- робота з формою (відкриття, закриття, зміна фільтра; оновлення даних, друк форми);
- робота зі звітом (друк, перегляд, відправка, висновок в файл), додатком (запуск програми, вихід з Додатка, запуск Word, Excel, блокнота);
- запуск запиту, макросу, друк таблиці, набір номера.

Користувач може сам програмувати макроси і процедури VBA для обробки різних подій, що виникають при роботі в формі.

Таким чином, форма є головним об'єктом, за допомогою якого користувач може спілкуватися з базою даних (застосуванням), тому потрібно приділити серйозну увагу цьому об'єкту бази даних, щоб користувачу будь-якої кваліфікації було легко працювати з базою даних. Призначення всіх елементів керування форми має бути зрозумілим. Якщо існують якісь особливості використання, то потрібно скористатися створенням підказок.

## 6.1. Створення форми

Форму можна створювати як з використанням **Мастера форм** так і у режимі **Конструктора**, з подальшою її модифікацією (додавання чи вилучення деяких елементів). При створенні нової однотабличної форми визначається таблиця БД, на основі якої створюється форма, вибираються поля таблиці, які повинні бути представлені в формі, здійснюється їх розміщення в макеті форми, створюються обчислювані поля і інші графічні елементи.

При редагуванні раніше створеної форми необхідно перейти в режим конструктора форм **Конструктор**. У цьому режимі можна виконувати всі зазначені вище дії для створення нових елементів, а також видалення елементів форми і їх модифікації.

### 6.1.1. Створення шаблона форми за допомогою Майстра форм

Для створення форми необхідно виділити таблицю, наприклад, **Лічильники_запити** та вибрати опцію головного меню **Создание,** після чого

обрати один з варіантів створення форм, представлених на закладинці **Формы** наступними піктограмами.

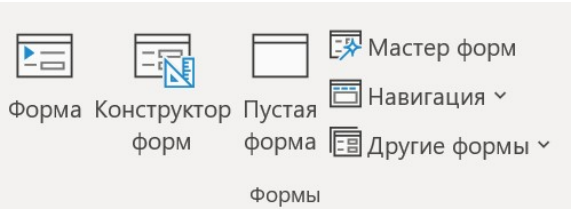



Рис. 6.2. Вікно вибору таблиці та її полів для форми

Можливості створення кожного з варіантів описані в підказках, які відкриваються при наведенні курсора на відповідну піктограму.

Якщо обрати піктограму **Мастер форм**, то на екрані з'явиться вікно **Создание форм** (рис. 6.2), у випадаючому списку якого бачимо ім’я вибраної раніше таблиці та поля які будуть використовуватися у формі. Можна перенести у вікно **Выбранные поля** всі поля таблиці або тільки деякі. Це залежить від завдання, яке буде реалізовуватися за допомогою цієї форми та натиснути кнопку **Далее**.

Після натискання кнопки **Далее** на екрані відкривається вікно вибору зовнішнього вигляду екранної форми - шаблону форми (рис. 6.3).

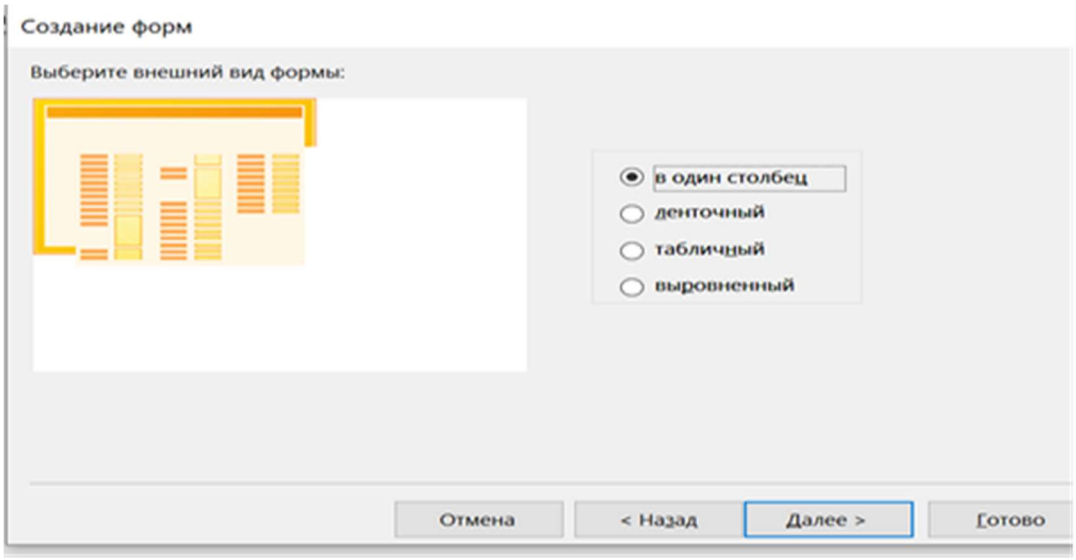
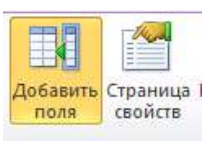



Рис. 6.3. Вікно вибору зовнішнього вигляду екранної форми

Для додавання додаткових полів є кнопка **Добавить поля** 

, яка відкриває список обраної таблиці **Список полей** (рис. 6.4).

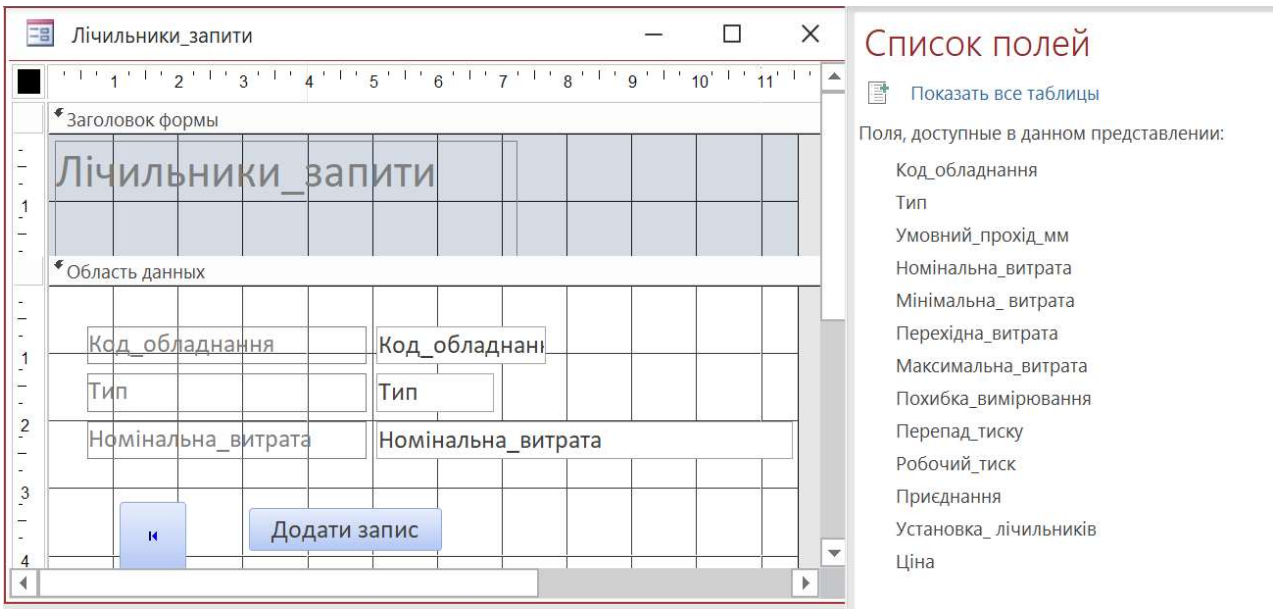



Рис. 6.4. Вікно **«Список полей» із с**писком полів обраної таблиці

При виборі шаблону **в один столбец** відкривається готова форма, вигляд якої відповідає ораному варіанту шаблона. Створену у такий спосіб форму можна відкрити у режимі конструктора (рис. 6.5) і редагувати її. В формі можна перетягувати поля і надписи і розташовувати їх у відповідності замовленого зразка, змінювати шрифт надписів, фон форми, додавати елементи керування і ,таким чином, кардинально змінювати вигляд форми.

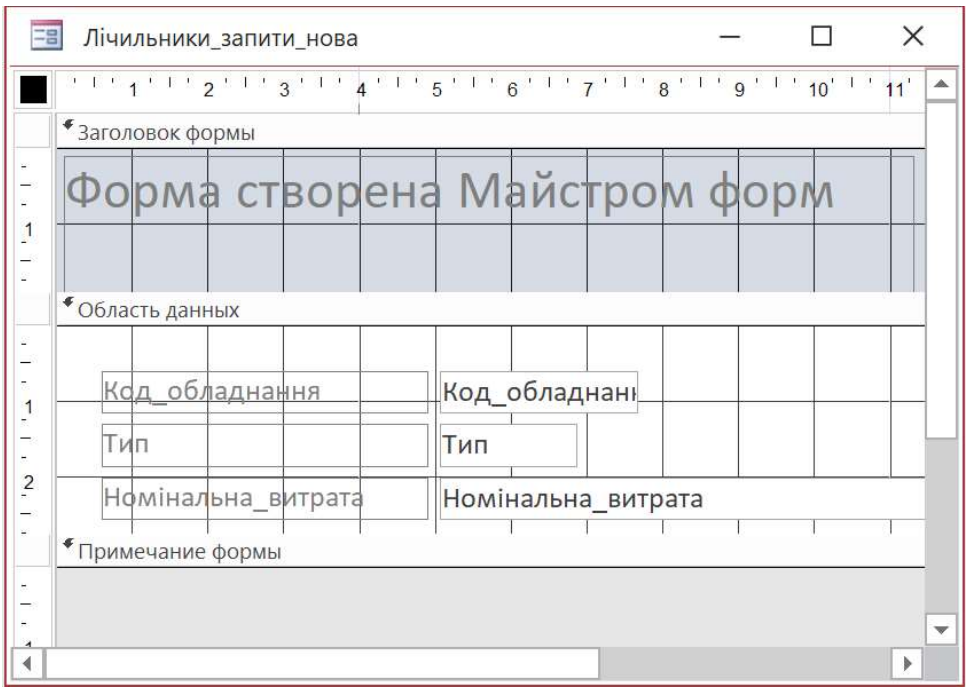



Рис. 6.5. Вигляд форми у режимі конструктора для шаблону **в один столбец**

З панелі **Элементы управления** (рис 6.6), на якій розташовані піктограми елементів керування (кнопка, поле вводу, надпис, малюнок і т.і.), можна перетягти на форму будь-який з них. Для того, щоб елементи керування виконували якусь дію, потрібно запрограмувати у властивостях цього елемента обрану подію, наприклад, **Нажатие кнопки, Двойное нажатие кнопки, Получение фокуса** і т.і. Доступ до властивостей елемента керування, можна отримати обравши опцію **Свойства** контекстного меню або обрати піктограму **Свойства** головного меню. Запрограмувати подію елемента керування можна використовуючи мову програмування VBA (Visual Basic for Application) або об'єкт бази даних – макрос. В ACCESS існує велика множина вбудованих елементів керування, події яких вже запрограмовані. Для цього необхідно перетягти потрібний елемент керування на форму, наприклад, кнопку. Відкриється вікно (рис. 6.7), в якому можна обрати дію, яка буде виконуватися при натисненні цієї кнопки. Для того щоб з'явилося вікно (рис. 6.7), необхідно активізувати наступну піктограму. 

. Для створення елементів керування з нестандартними діями, навпаки, ця піктограма має бути неактивною.

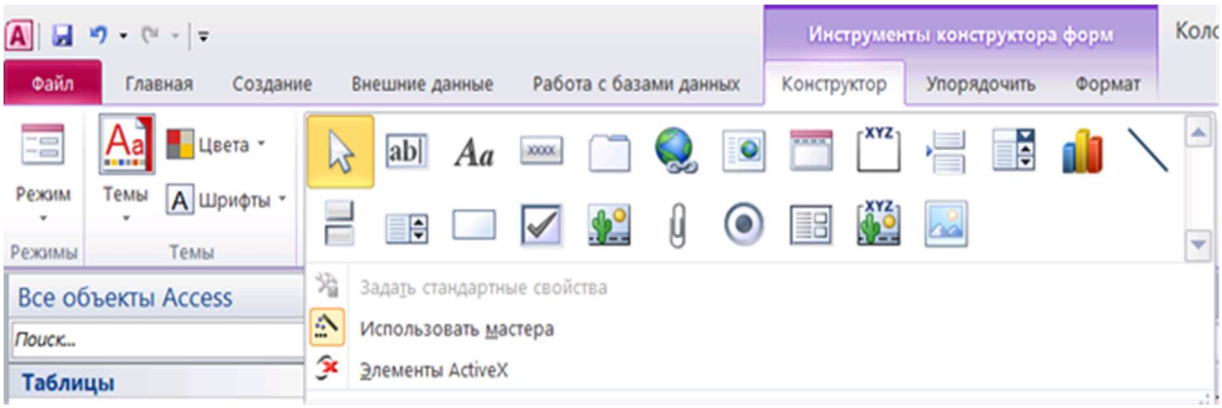



Рис. 6.6. Розгорнутий список елементів керування

### 6.1.2. Створення форми в режимі Конструктор форм

Для конструювання форм в Access використовується **Конструктор форм**. Режим конструктора форм можна використовувати як для створення нової форми, так і для редагування раніше створеної форми. Найчастіше цей режим

використовується для створення форм не зв’язаних з таблицями даних та для редагування форм, які попередньо були створені.

Форма в режимі Конструктор форм включає наступні розділи.

• Заголовок форми – область у верхній частині форми, в якій можна розташовувати інформацію, яка не залежить від змісту записів, наприклад, назву форми, назву завдання, яке можна виконати створеною формою.

• Область даних – представляє собою центральну частину форми, на якій можуть бути розташовані поля з даними таблиць або запитів, надписи, різноманітні елементи керування.

• Примітка форми – область у нижній частині форми, в яку можна виводити інформацію у вигляді пояснень.

### 6.1.3. Створення кнопок переходу по записам та кнопок додавання даних

Для створення кнопок переходу по записам необхідно послідовно виконати наступні дії.

Розмістити на формі елемент керування **Кнопка**. Після чого з'явиться вікно настройки кнопки (рис. 6.7).

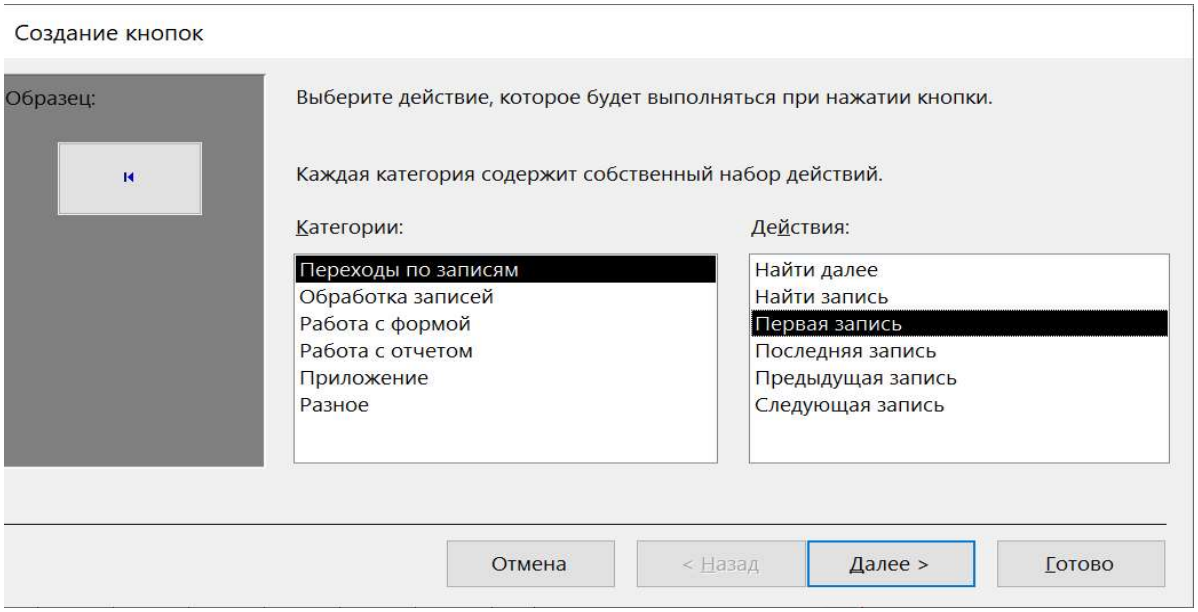



Рис. 6.7. Вигляд вікна **Создание кнопок**

В полі **Категории** потрібно вибрати **Переходы по записям**, а в полі **Действия - Первая запись**. Натиснути кнопку **Далее** та вибрати малюнок для

кнопки. Натиснути кнопку **Готово**. Аналогічні дії виконати для створення кнопок **Предыдущая запись, Следующая запись** і **Последняя запись**, **Закрыть форму** (рис. 6.8). Кожна кнопка виконує одну дію, наприклад, перейти на перший запис виконує стандартна кнопка **Первая запись**. На створеній формі відображаються дані одного запису таблиці. Кнопки переходу використовуються для перегляду першого, попереднього, наступного чи останнього запису. На рис. 6.8 показана форма із створеними кнопками.

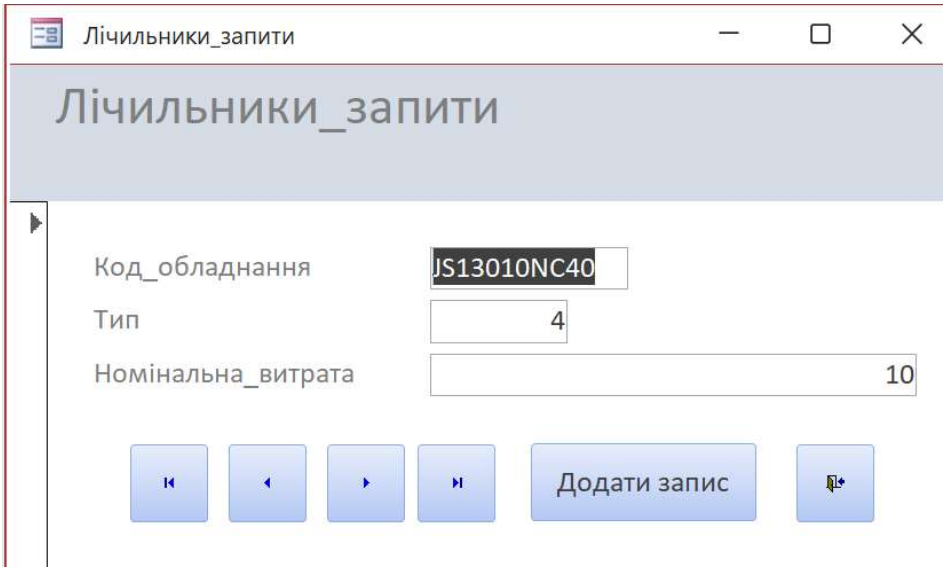



Рис. 6.8. Вигляд форми з кнопками **переходу по записам** в режимі перегляду даних

Розмістити на формі елемент керування **Кнопка**. В полі **Категории** вибрати **Обработка записей,** а в полі **Действия** вибрати **Добавить запись**. Натиснути кнопку **Далее** та ввести назву кнопки "**Додати запис**" в полі вводу **Текст**. Натиснути кнопку **Готово** (рис. 6.9).

Перехід до першого запису таблиці, що відбувається після події натиснення кнопки, реалізований процедурою VBA. Процедури VBA стандартних елементів керування можна редагувати. Розглянемо приклад редагування процедури стандартної кнопки Добавить запись . Спочатку розміщуємо кнопку на формі (вона отримала ім’я Кнопка11), а потім з вікна властивостей кнопки викликаємо редактор VBA.

Змінимо підпис кнопки **Кнопка11** на **Додати запис**. Потім у вікні властивостей кнопки **Кнопка11** в рядку події **Нажатие кнопки** потрібно натиснути кнопку з трьома крапками (рис. 6.9).

У вікні редактора кода VBA буде створений шаблон процедури.

***Private Sub Кнопка11_Click()***

***…….***

***End Sub***

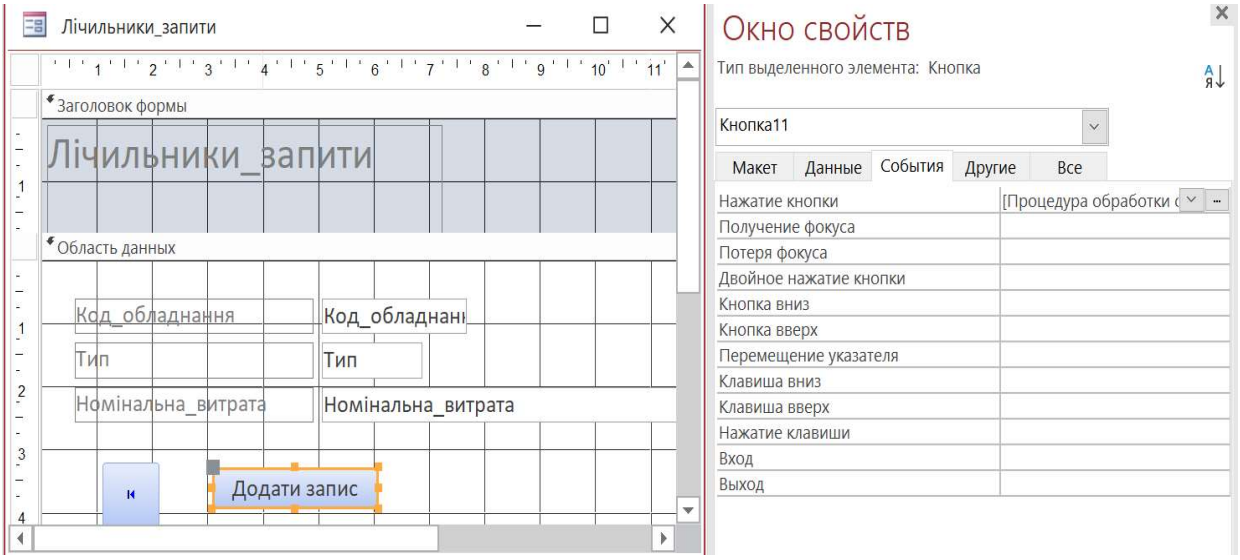



Рис. 6.9. Вигляд форми з вікном властивостей кнопки Додати запис

В тіло процедури потрібно вставити наступний код.

***On Error GoTo Err_Кнопка11_Click***

***MsgBox "Додати запис?", vbYesNo + vbExclamation, "Дайте відповідь"***

***DoCmd.GoToRecord , , acNewRec***

***Exit_Кнопка11_Click:***

***Exit Sub***

***Err_Кнопка11_Click:***

***MsgBox Err.Description***

***Resume Exit_Кнопка11_Click***

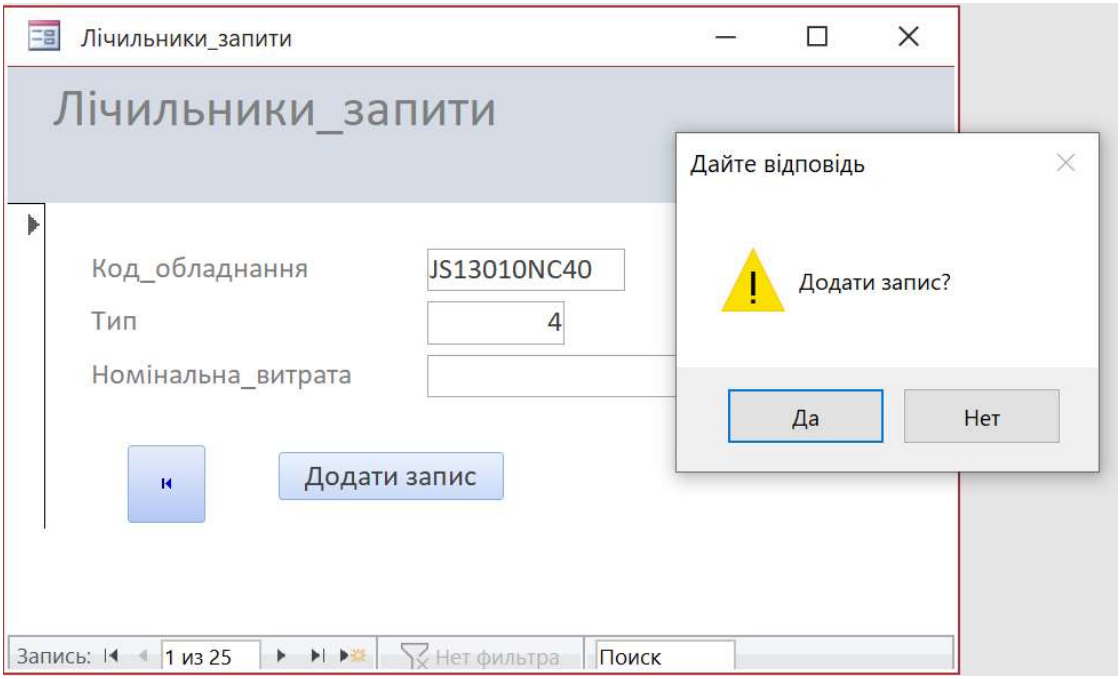



Рис. 6.10. Вигляд форми після натиснення кнопки Додати запис

До тексту стандартної процедури ми додали функцію MsgBox " Додати запис?", vbYesNo + vbExclamation, " Дайте відповідь " яка виводитиме на екран діалогове вікно з текстом «**Додати запис**?», а змінна vbYesNo виводить в це ж вікно кнопки **Да** та **Нет**.

Вигляд форми після натиснення кнопки Додати запис представлено на рис. 6.10. Вибір кнопок Да або **Нет** в цьому прикладі буде приводити до одного і того ж результату – буде відкриватися порожня форма для введення нових даних. Для того щоб форма відкривалася в режимі додавання даних лише при виборі кнопки **Да**, потрібно додати наступні оператори VBA

```
answer = MsgBox("Додати запис?", vbYesNo)
 If answer = vbYes Then
 DoCmd.GoToRecord , , acNewRec
 End If
```

Після редагування процедура Кнопки11 для додавання даних до таблиці Лічильники_запити матиме вигляд:

```
Private Sub Кнопка11_Click()
Dim answer As Integer
On Error GoTo Err_Кнопка11_Click
 answer = MsgBox("Додати запис?", vbYesNo)
 If answer = vbYes Then
```

```
 DoCmd.GoToRecord , , acNewRec
 End If
Exit_Кнопка11_Click:
 Exit Sub
Err_Кнопка11_Click:
 MsgBox Err.Description
 Resume Exit_Кнопка11_Click
End Sub
```

Ми розглянули перші приклади використання мови VBA для реалізації подій елементів керування. Більш детально будемо вивчати можливості мови VBA для реалізації складних конструкцій форм пізніше.

І на завершення роботи з цією формою, необхідно змінити стиль створеної форми. Для цього в опції Свойства Формы в полі Область выделения –вибрати **Нет**, Кнопки перехода – **Нет**, Разделительные линии – **Нет** (рис. 6.8).

### 6.1.3. Додавання у форму надписів

Надпис являє собою елемент, що відображає статичний текст (тобто текст, який користувач у формі змінити або скопіювати не може). Найчастіше надписи використовуються для відображення назв полів, можуть застосовуватися і для:

- відображення назви вільного елемента;
- відображення заголовка або підзаголовка форми;
- відображення інструкцій з рекомендаціями по заповненню полів форми.

### 6.1.4. Вирівнювання полів і надписів

Щоб вирівняти поля, надписи, кнопки, потрібно натиснути праву кнопку миші та вибрати параметри вирівнювання у опцію контекстного меню **Выровнять** (рис. 6.11).

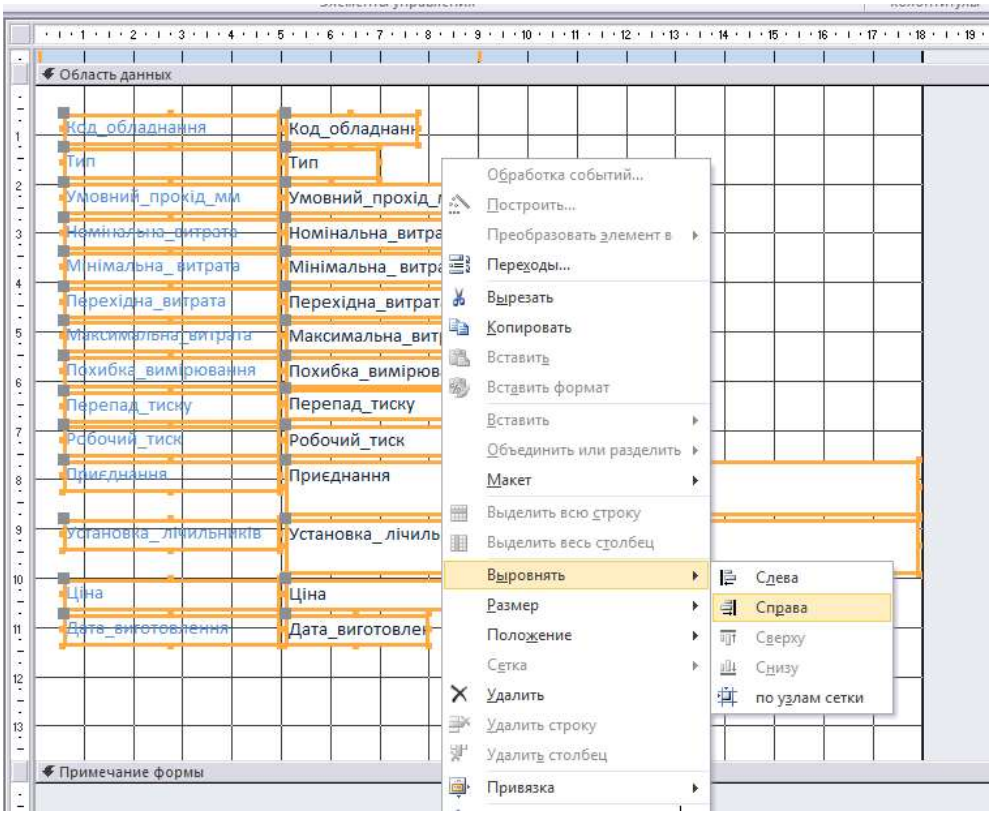



Рис. 6.11. Вирівнювання елементів керування форми.

## 6.2. Маски введення даних

Маска вводу – це рядок символів, який задає формат припустимих значень для введення. Маски вводу можна використовувати в полях таблиці, запитів і елементів керування форм і звітів. Маска вводу зберігається як є властивість об'єкта.

Маска вводу використовується у випадках, коли важливо забезпечити дотримання формату введених значень. Наприклад, маскою вводу можна скористатися для поля, у якому зберігаються номери телефонів, щоб програма Access вимагала введення десяти чисел. Якщо який-небудь користувач введе номер телефону без коду населеного пункту, програма Access не записуватиме дані, доки не буде додано код населеного пункту [3].

### 6.2.1. Використання майстра масок введення

Найпростішим способом створення масок є використання програми майстра. Для чого потрібно виконати наступні дії.

- Відкрити вікно властивостей поля, формат даних якого потрібно контролювати.
- Відкрити вкладку **Данные**.
- Клацнути в рядку Маска ввода.
- Клацнути на кнопці (три крапки) поруч із вибраною коміркою (рис. 6.12).

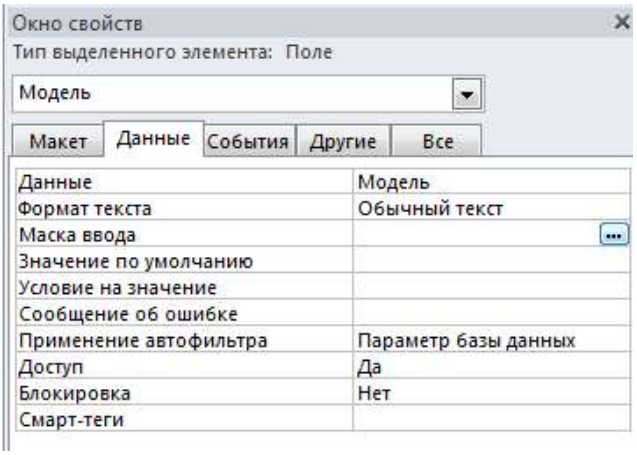



Рис. 6.12. Вікно властивостей поля для створення маски вводу

- У списку масок уведення вибрати потрібну (або близьку до бажаної) і натиснути кнопку **Далее** (рис. 6.13).

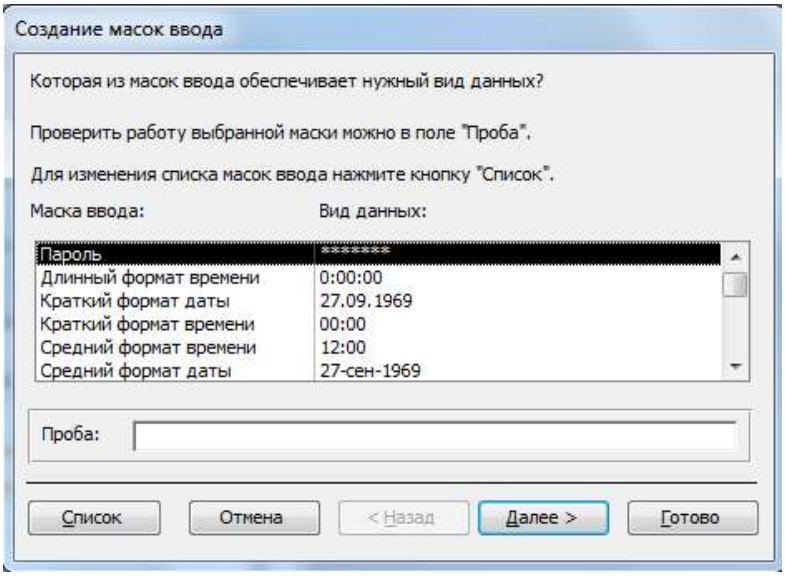



Рис. 6.13. Вікно створення маски з вбудованими форматами для введення даних

Для внесення змін у маску використовується поле **Маска ввода.** При цьому для вибору знаків можна використовувати список **Знаки заполнителя**. По завершенню роботи клацніть на кнопці **Далее**.

- Після цього натисніть кнопку **Готово**.

### 6.2.2. Створення маски для введення номера телефону

Іншим способом створення шаблону є створення маски для введення вручну. Для чого потрібно виконати наступні дії.

- Відкрити вікно властивостей поля, формат даних якого потрібно контролювати.
- Відкрити вкладку **Данные**.
- Клацнути в рядку **Маска ввода** та набрати з клавіатури дозволені символи, наприклад, для створення маски введення пароля *******, де символ * означає дозвіл на введення будь-якого символу (рис. 6.14).

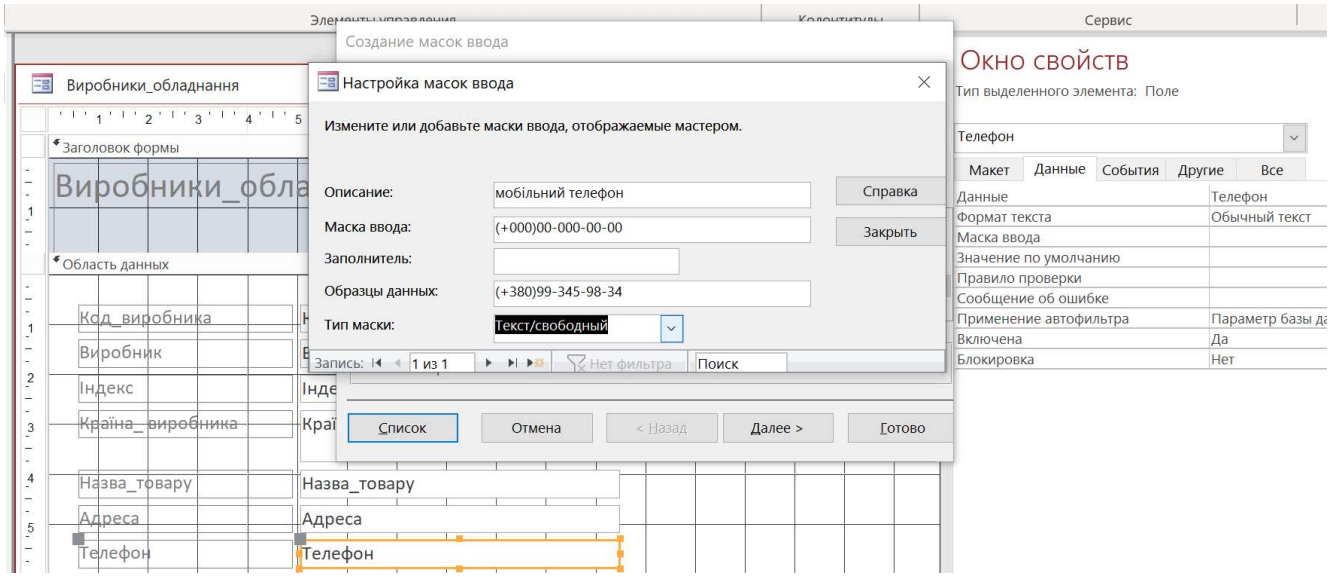



Рис. 6.14. Вікно створення зразка маски вводу номера телефона

Після закінчення набору, Access модифікує створену маску введення вигляд, представлений на рис. 6.15.

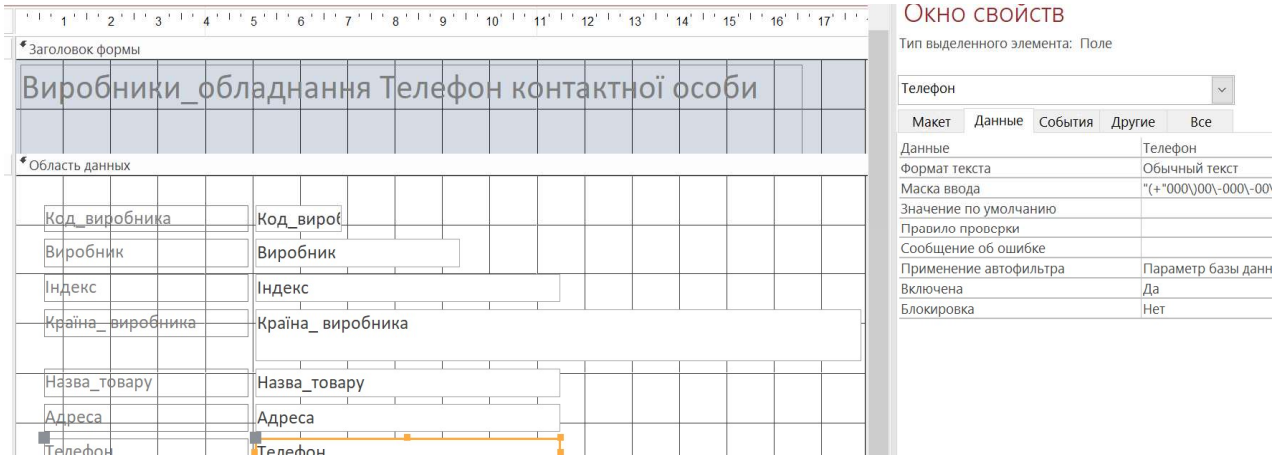



Рис. 6.15. Вигляд створеної маски модифікованої Access у вікні властивостей

В якості символу заповнювача вибрано символ підкреслення. При додаванні даних про нового відповідального в полі Телефон буде виводитися маска вводу (рис. 6.16). Таким чином, всі дані про телефон відповідального від фірми-виробника буде вводитися в однаковому вигляді, що важливо для наповнення бази вірними однотиповими даними.

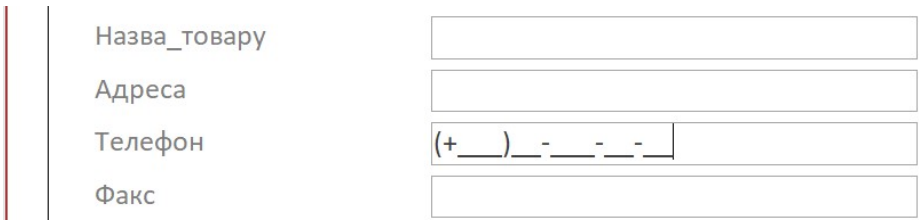



Рис. 6.16. Вигляд створеної маски при введенні нових даних

Для вибору виведення назви грошових одиниць можна скористатися шаблоном, представленим у рядку Формат поля властивостей поля Ціна - #,# “grn”. Та обрати в рядку Число десятичных знаков – 2. Символ # - означає виведення будь-якої кількості знаків. А значення 2 – означає що після коми буде виводитися два числових знаки.

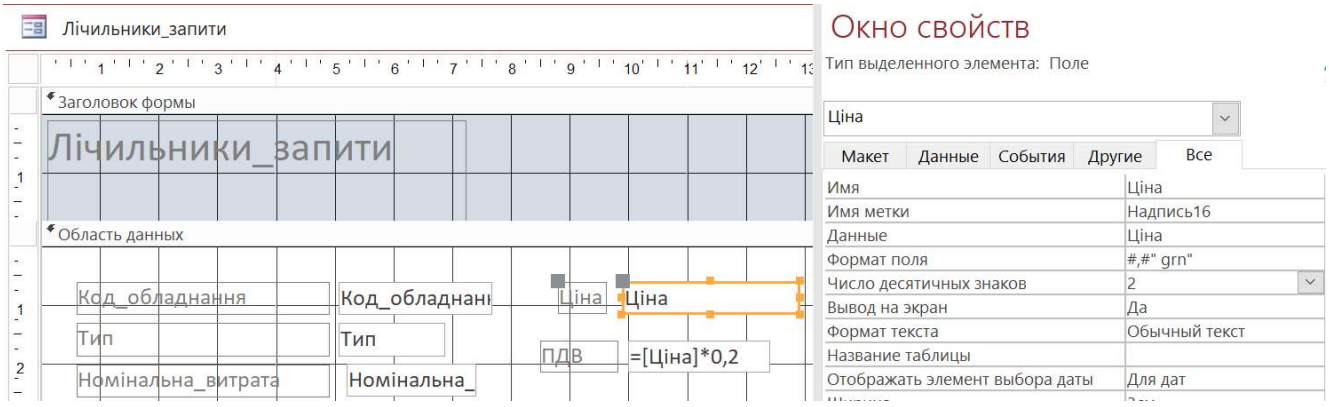



Рис. 6.17. Вікно властивостей встановлення формату виведення назви грошових одиниць

## 6.3. Створення поля, значення якого обчислюється

В формах, звітах та запитах можна створювати поля, значення яких обчислюються з використанням відповідних виразів. Вирази можуть включати змінні, сталі, вбудовані функції та оператори, які виконують певні операції над перерахованими об’єктами. Записати вираз можна як безпосередньо у полі **Данные** (форми або звіта), так у відповідному рядку властивостей поля або у вікні **Построителя выражений**. Відкрити вікно **Построителя выражений**

можна скориставшися піктограмою , або використати спосіб , який описано далі. .

Для створення поля, значення якого обчислюється необхідно розмістити на формі елемент керування **Поле** з панелі елементів (рис. 6.18**).**


Рис. 6.18. Вікно Панель елементів для вибору нового поля

Відкрити вікно **Свойства** новоствореного поля **Поле17**(рис. 6.19**)**.


Рис. 6.19. Фрагмент форми Лічильники запити з новостворенним полем

Новостворене поле отримало системне ім’я **Поле17** (рис. 6.19). Це поле має статус вільного поля (свободный). Тобто поле не зв’язане з ні одним полем таблиці, яка є джерелом даних даної форми. А це означає, що дані поля не будуть запам’ятовуватися, а лише виводитися у форму для перегляду. Але для того щоб дані в полі з’явилися, потрібно створити вираз, результат якого можна буде в ньому побачити.

У новостворенному полі будемо обчислювати податок ПДВ, який складає 20% від ціни. Вираз у полі Построителя выражений можна набрати за допомогою клавіатури. Але доцільніше скористатися можливостями набору, які надає Построитель выражений. Для цього в полі <Таблицы> знайдемо таблицю Лічильники_запити та натиснемо на полі Ціна два рази. Потім за допомогою операторів обчислення введемо вираз “=[ Ціна]*0,2” . Натискаємо ОК.

В полі **Данные** натиснути кнопку після цього відкриється **Построитель выражений**.

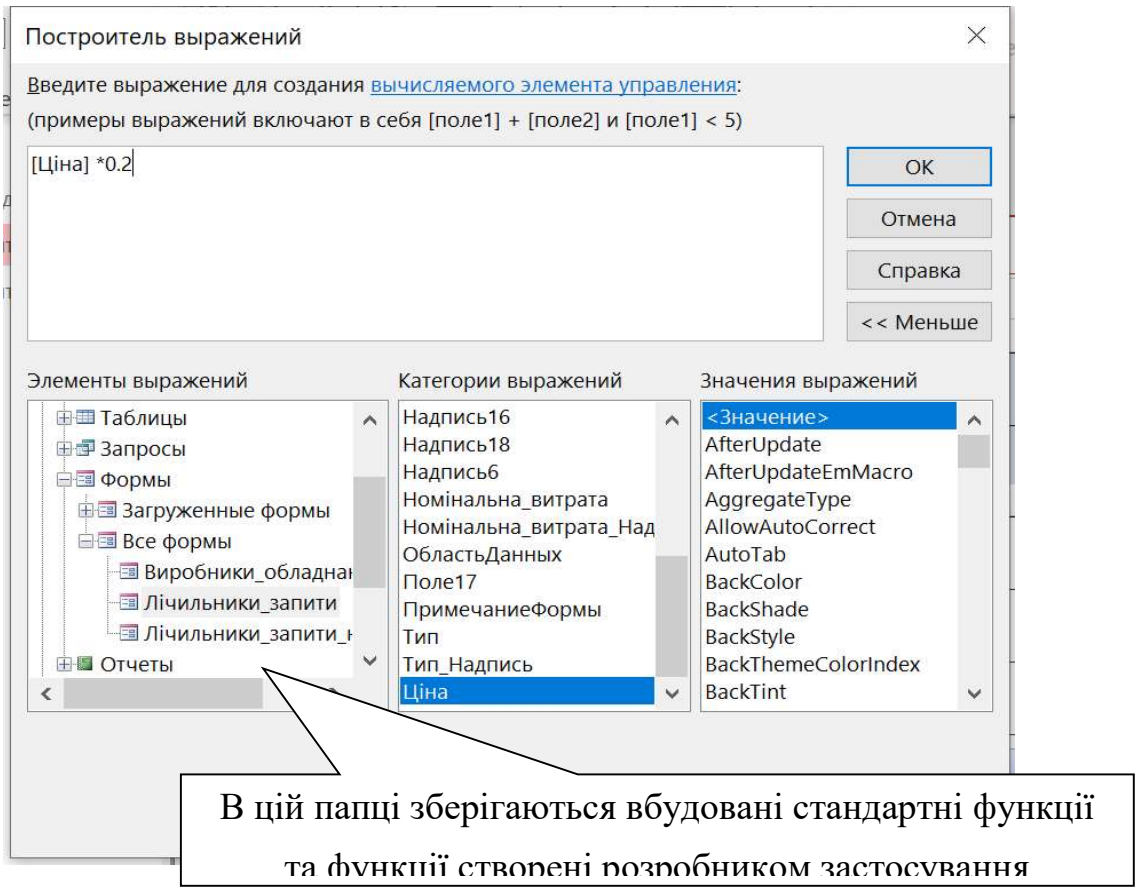



Рис. 6.20. Вікно **Построитель выражений** для запису формули Поля17

У формі в полі замість Свободный з’являється наш вираз. Цей вираз можна записати безпосередньо у полі **Свободный** нашої екранної форми.

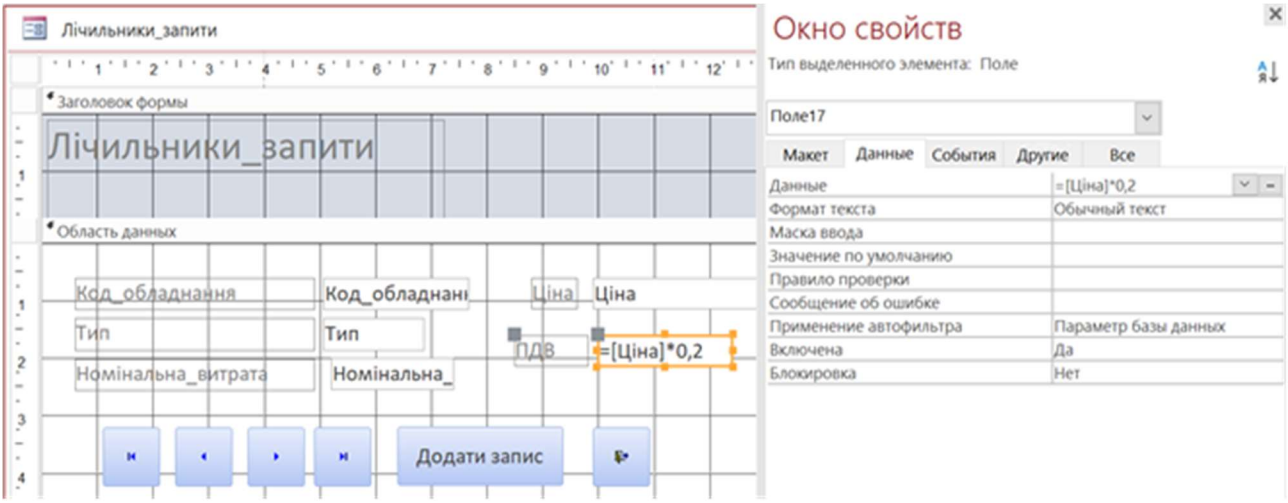



Рис. 6.21. Вигляд форми з властивостями поля Поле17 з виразом обчислення ПДВ

## 6.3. Реалізація обробки подій елементів керування макросом

У Microsoft Access для реалізації подій елементів керування застосовуються макроси. Макроси в свою чергу складаються з макрокоманд. Макрокомандою можна відкрити форму, звіт, надрукувати звіт, запустити на виконання запит, застосувати фільтр, привласнити значення, створити користувальницьке меню або панель команд.

Макрокоманда ВыполнитьКоманду (RunCommand) дозволяє виконати будь-яку вбудовану команду Access, що виводиться в меню, на панелі інструментів або в контекстному меню. Наявний в Access набір макрокоманд реалізує практично будь-які дії, які необхідні при розробці невеликих персональних застосувань користувача.

Подія натискання на будь-яку кнопку форми **Головна форма** має виконувати попередньо створений макрос для реалізації відповідного завдання комп'ютерного практикуму.

### 6.3.1. Створення простого макросу

В СУБД Microsoft Access, починаючи з версії Access 2010, було вдосконалено конструктор макросів.

Конструктор макросів дозволяє створювати більш гнучкі, зрозумілі і зручні в використанні макроси. Створені макроси можна прив'язувати до таблиць, після чого кожен об'єкт, який створюється на основі відповідної таблиці, буде успадковувати зв'язаний з таблицею даних макрос. Конструктор виразів (Построитель выражений) тепер підтримує технологію IntelliSense, що спрощує створення виразів. Для цього в конструкторі застосовуються списки, що розкриваються, дозволене повторне використання існуючих макросів, операція перетягування, а також копіювання і вставка через буфер обміну [1].

Щоб почати створення нового макросу необхідно у вікні бази даних обрати опцію меню **Создание** натиснути на об'єкт Макрос і натиснути кнопку **Создать.** Після чого з'явиться вікно (рис. 6.22).


Рис. 6.22. Вибір конструктора створення макросів

Після вибору піктограми 

відкриється вікно конструктора макросів (рис. 6.23).

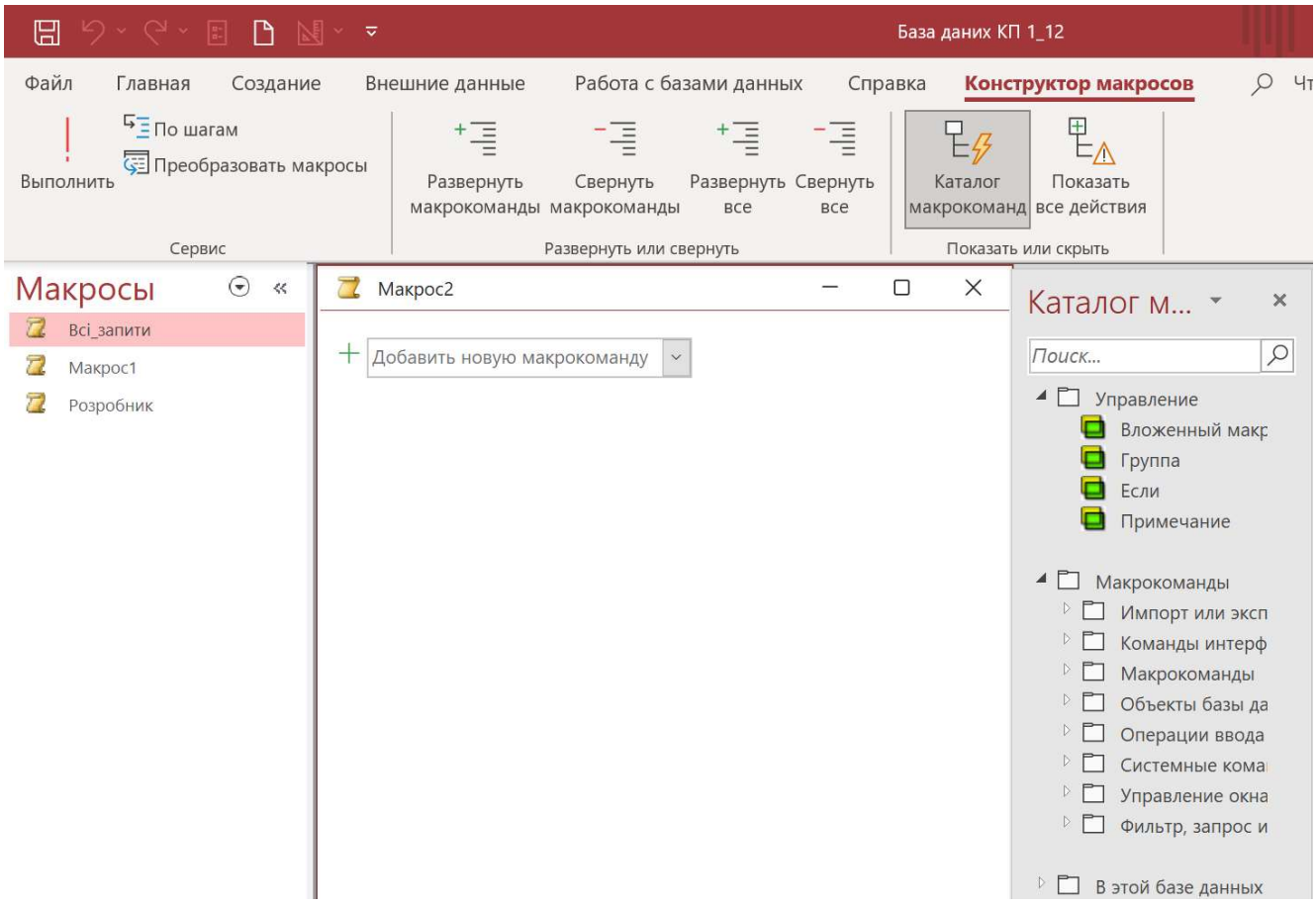



Рис. 6.23. Вигляд конструктора для створення макросів

Після завершення вибору макрокоманд макросу, його необхідно зберегти, а потім приєднати до обраної події відповідного об’єкту (рис. 6.24, 6.25).

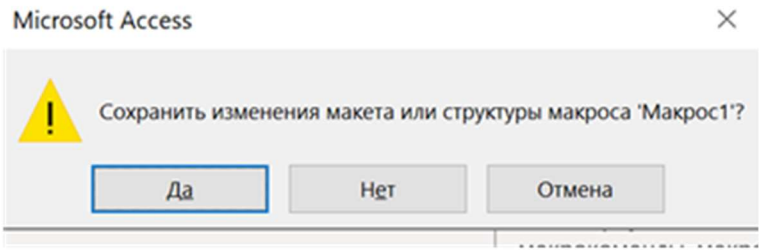



Рис. 6.24. Вигляд вікна з пропозицією збереження створеного макросу

Для того, щоб макрос зв’язати з відповідною кнопкою необхідно:

- Викликати **Свойства об’єкта** (кнопки).

У вкладці **События** до події **Нажатие** кнопки обрати ім’я необхідного макросу із розкритого списку (рис. 6.25).

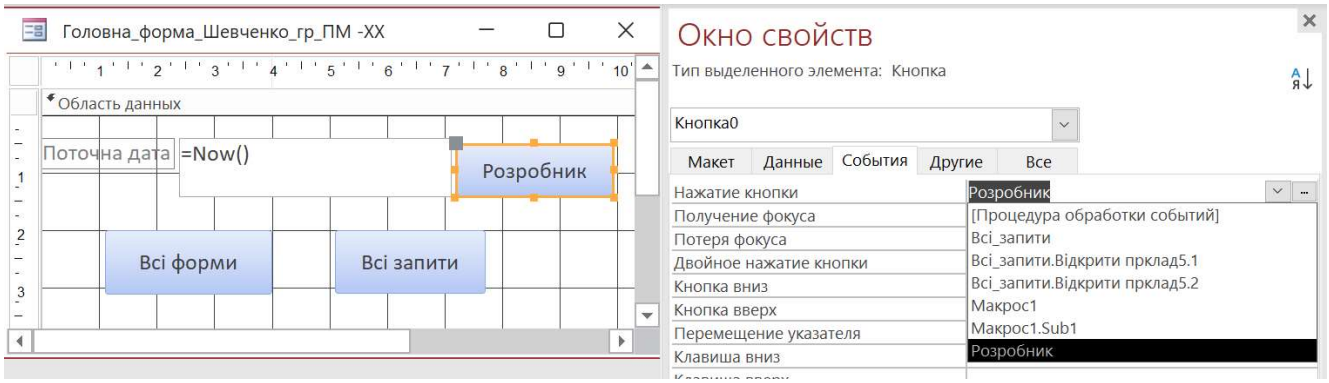



Рис. 6.25. Приєднання макросу до події **Нажатие кнопки** кнопки Розробник

Головну форму, представлену на рис. 6.25, потрібно буде доопрацьовувати на наступних комп’ютерних практикумах у відповідності до сформульованих там задач. Ця форма має відкриватися при виклику створеної бази даних на виконання.

### 6.3.2. Створення макросу для виведення інформації про розробника

Після вибору макрокоманди, такої як Окно Сообщения, Access виведе наступний шаблон макрокоманди (рис. 6.26).

Значення аргументу Сообщение являє собою текст, який Access буде виводити у вікні повідомлення. Аргумент Сигнал задає, чи буде звуковий сигнал супроводжувати появу інформаційного вікна. Аргумент Тип дозволяє обрати з випадаючого списку (рис. 6.26) вигляд вікна з повідомленням. В аргумент Заголовок вводиться текст, що з'явиться в заголовку вікна повідомлення.

Вікно повідомлення представляє собою один з можливих варіантів форми Access, а саме модальної форми. Для закриття модальної форми необхідно натиснути кнопку ОК. Поки форма відкрита – вона є пріоритетною не дає можливості виконувати інші завдання.

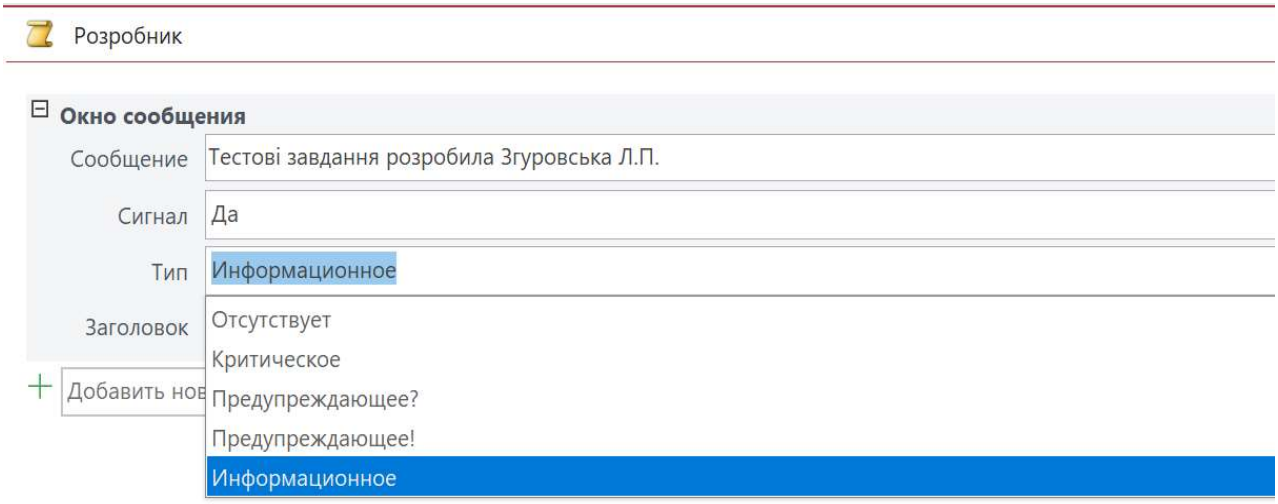



Рис. 6.26. Вигляд макроса Розробник у режимі конструктора

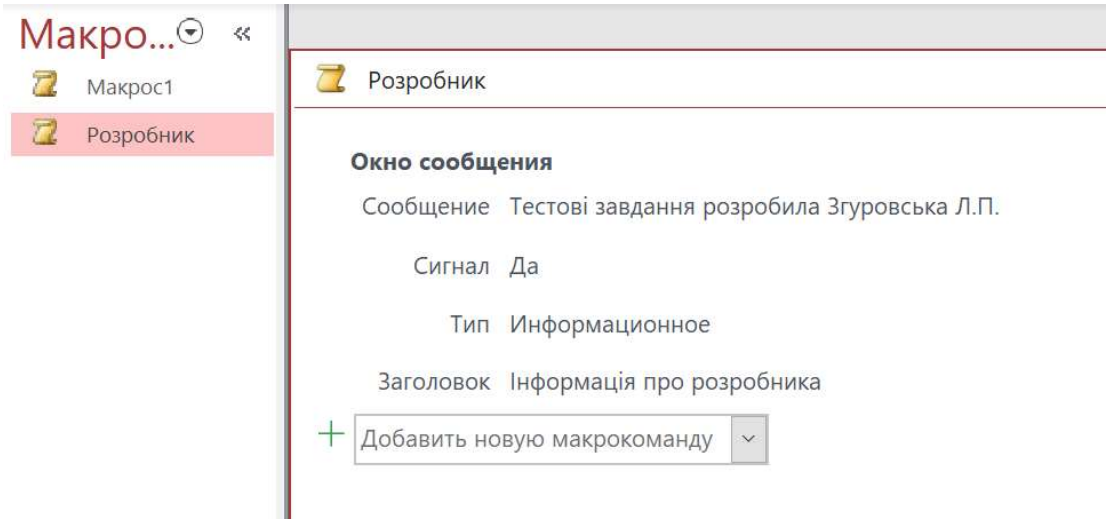



Рис. 6.27. Вигляд макроса Розробник у режимі конструктора

В результаті після натискання кнопки Розробник буде виводитися наступне інформаційне вікно

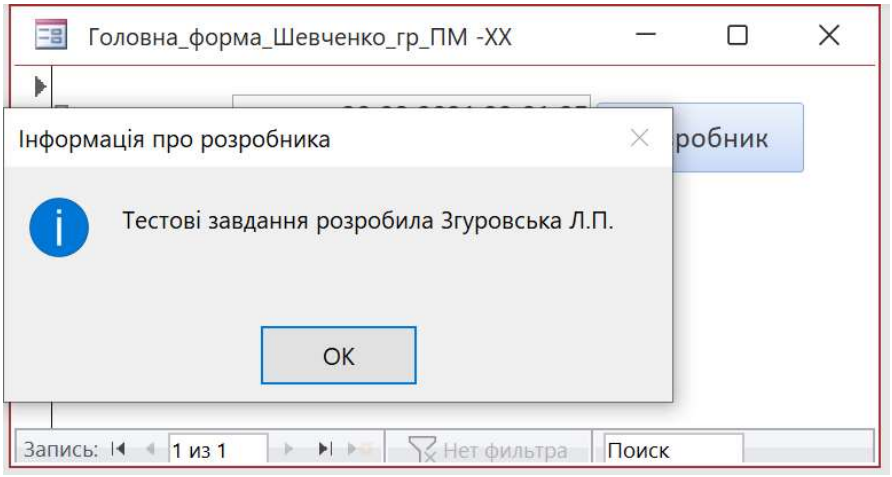



Рис. 6.28. Вигляд інформаційного вікна, створеного макросом Розробник

## 6.4. Завдання до комп'ютерного практикуму

1. Для виконання комп’ютерного практикуму 6 і всіх наступних (крім КП10) продовжуємо використовувати базу даних створену на комп’ютерних практикумах 1,4-5.
2. В результаті виконання комп’ютерного практикуму 6 мають бути створені форми для завдань 4-12, та дві форми з кнопками для виконання завдань запитів (таблиця 6.1) та форма для виклику форм (завдання 4-10).
3. В структуру таблиць ЛічильникиN_FFF (для варіантів 4,9,13,16), КлапаниN_FFF (для варіантів 1,8,11,17), КраниN_FFF (для варіантів 6,10,14,18), ЗаслінкиN_FFF (для варіантів 3,12,15,19), ФільтриN_FFF (для варіантів 2,5,7,20) додати поле ціна та Дата_виготовлення. Заповнити ці поля даними з таблиці Склад_обладнання.
4. Створити форму з кнопками (приклад кнопок на рис. 6.8.) для таблиці ЛічильникиN_FFF (для варіантів 4, 9, 13, 16), для таблиці КлапаниN_FFF (для варіантів 1, 8, 11, 17), для таблиці КраниN_FFF (для варіантів 6, 10, 14, 18), для таблиці ЗаслінкиN_FFF (для варіантів 3, 12, 15, 19), для таблиці ФільтриN_FFF (для варіантів 2, 5, 7, 20) за допомогою Майстра форм.

**УВАГА!** Поля всіх створюваних форм повинні відповідати наступним вимогам:

- При введенні даних, значення даних мають автоматично перевірятися ( за допомогою виразів або масок вводу даних ) і при

виявленні помилок необхідно передбачити виведення інформації про помилку, що сталася.

- У полях вводу даних мають використовуватися текстові коментарі, які пояснюватимуть користувачеві особливості введення даних у відповідне поле.
- Для кнопки **Додати запис** вставити виведення повідомлення у відповідну процедуру.

5. Створити у формі індивідуального варіанту **свободное** поле **Дата и время,** інформація до якого має виводитися функцією NOW.
6. Обчислити значення поля **ПДВ** для ціни обладнання.
7. Створити форму для таблиці **Склад_обладнання** за допомогою **Мастера**, розташувавши поля у стовпчик. Обчислити значення поля **Вартість** для кожного товару.
8. Створити форму для таблиці Назва_обладнання за допомогою Автоформа: ленточная.
9. В таблицю **Замовлення** додати поле Номер_замовлення та поле Телефон.
10. Створити форму для таблиці **Замовлення** за допомогою **Автоформа: в столбец. Значення полів типу Дата/Время** вводити використовуючи шаблон. Для поля Телефон створити маску вводу з урахуванням коду країни та оператора зв'язку.
11. Створити головну форму комп'ютерного практикуму №6 за зразком (рис. 6.29).

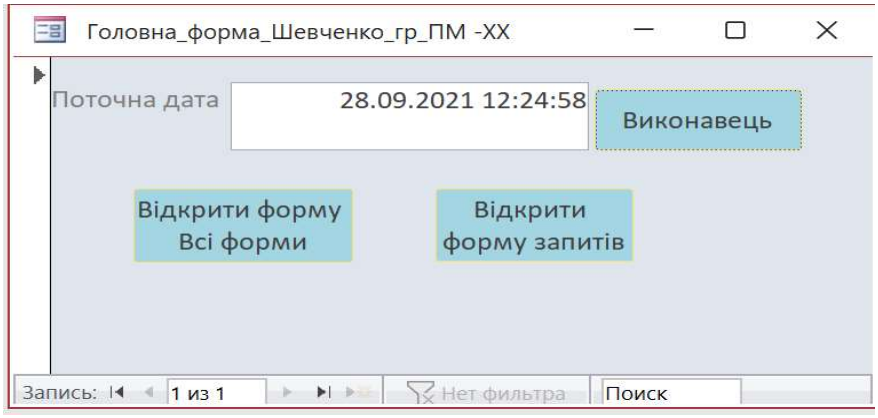



Рис. 6.29. Вигляд головної форми комп'ютерного практикуму 6

При натисканні кнопки **Виконавець** має відкриватися інформаційне вікно з інформацією про студента, який виконує практикум( Прізвище, Ім’я , шифр групи, семестр і рік). Кнопка **Відкрити форму Всі форми** відкриває форму другого рівня, на якій розташовані кнопки для відкриття усіх створених у попередніх завданнях форм. Кнопка **Відкрити форму запитів** відкриває форму другого рівня (рис. 6.32), на якій розташовані кнопки для виконання запитів, створених у комп’ютерних практикумах 4 і 5. Кнопки для запуску запитів на виконання повинні бути створені у відповідності до завдання індивідуально варіанту (табл. 6.1). Подія кожної кнопки має бути реалізована відповідним макросом. Наприклад, для кнопок Приклад 5.1 та Приклад 5.2 макрос виглядатиме так

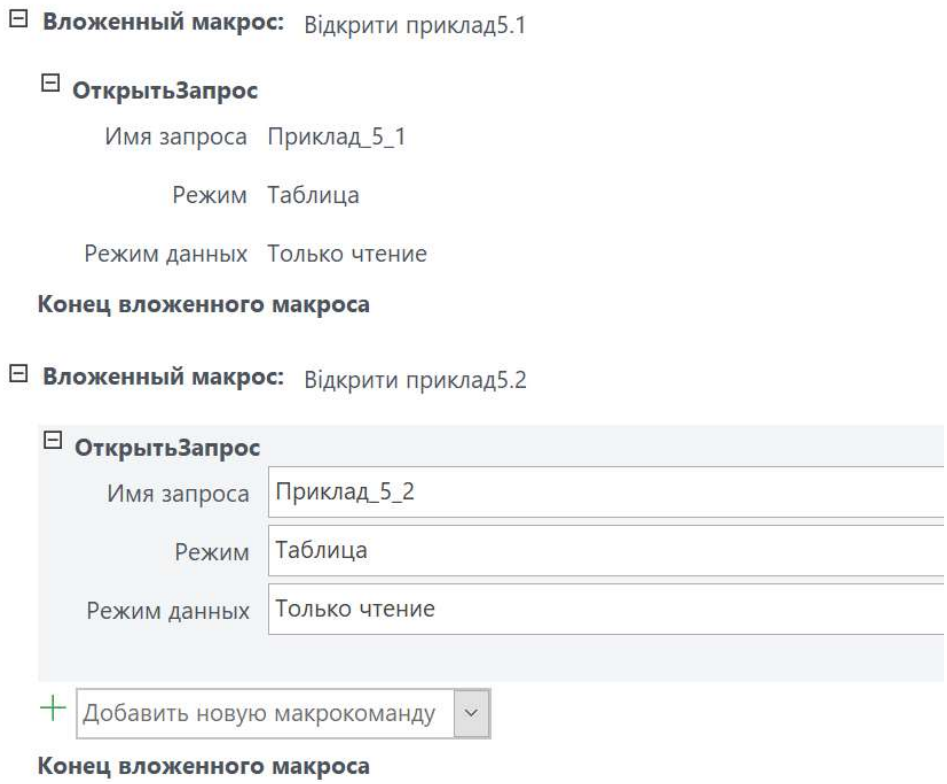



Рис. 6.30. Макрос Всі_запити з двома вкладеними макросами

В одному макросі можна зберігати декілька макросів для виконання однотипових дій. Для реалізації події в такому випадку необхідно підключити вкладений макрос (рис. 6.31).

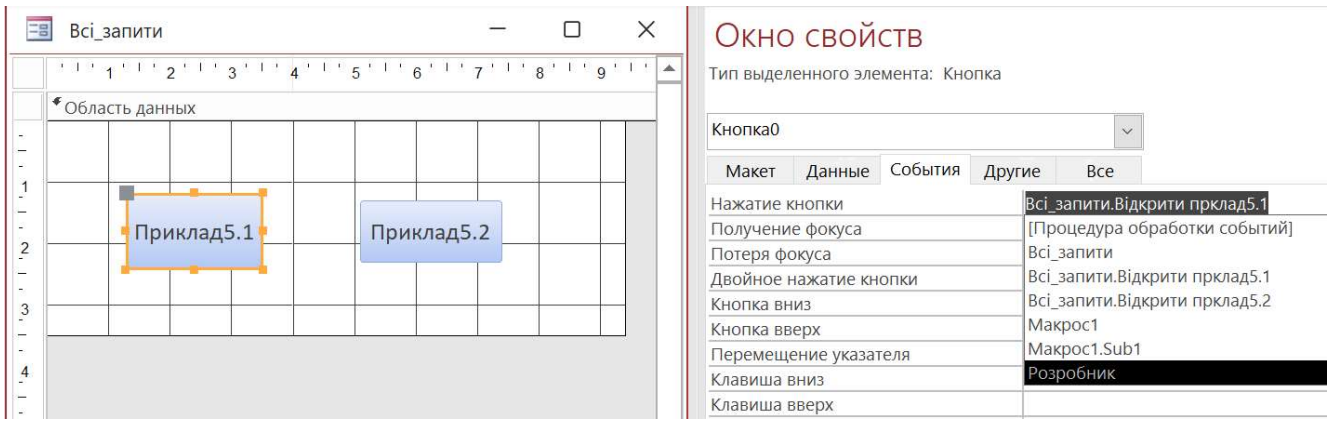



Рис. 6.31. Підключення макросу ВсіЗапити.Відкрити Приклад 5.1 до події кнопки Приклад 5.1

Форма другого рівня (рис. 6.32), на якій розташовані кнопки для виконання запитів, створених у комп'ютерних практикумах 4 і 5 має бути попередньо створена.

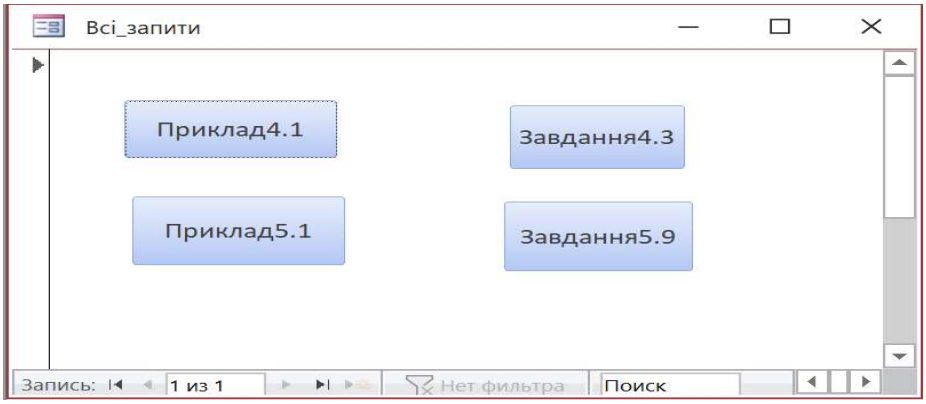



Рис. 6.32. Приклад вигляду форми другого рівня для запуску запитів на виконання

Таблиця 6.1 Запити для реалізації функціонування кнопок форми рис. 6.32

| Варіант | Запити | Варіант | Запити |
|---|---|---|---|
| 1 | Завдання 4.5,5.5,5.10<br>Приклад 5.5 | 9 | Завдання 5.6-5.9 |
| 2 | Приклади 4.1–4.3, Завдання 5.6 | 10 | Завдання 4.2,4.4, 5.2,5.5 |
| 3 | Завдання 4.3,4.5, 5.4,5.8 | 11 | Завдання 4.1,4.2,5.3,5.8 |
| 4 | Приклади 5.2, 5.5, Завдання 5.5,5.10 | 12 | Завдання 4.6-4.9 |
| 5 | Завдання 5.6-5.10 | 13 | Приклади 4.1,4.3, Завдання 5.6,5.10 |
| 6 | Приклади 5.1-5.4 | 14 | Завдання 4.1-4.4 |
| 7 | Завдання 4.4,4.8,5.5,5.9 | 15 | Завдання 5.1-5.4 |
| 8 | Приклади 5.5,4.3, Завдання 5.7,5.10 | 16 | Завдання 4.6,5.8,5.10<br>Приклад 5.5 |

## 6.5. Контрольні запитання

1. З яких областей складається макет форми в режимі Конструктора?
2. Яка область форми використовується для роботи з даними таблиці?
3. . В якому розділі властивостей форми визначаються основні параметри її виду?
4. Поясніть призначення області форми Заголовок форми
5. Поясніть призначення області форми Примечание.
6. Як змінити назву форми?
7. Як у формі встановити захист від зміни даних поля?
8. Чи може поле зі списком включати значення декількох полів запису?
9. Звідки може отримувати значення поле зі списком?
10. Як створити макрос для виведення даних про розробника бази даних?

# Комп'ютерний практикум № 7. Створення форм бази даних з об’єктами ActiveX

*Мета роботи* - набути навичок створення форм бази даних з автоматизацією введення даних.

**Теоретичні відомості**

Форма є головним об’єктом бази даних (застосування). Використовуючи форму користувач може спілкуватися з застосуванням, а саме: вводити нові дані, редагувати раніше введені дані, отримувати інформацію з бази даних для перегляду, аналізу та подальшого використання. Для виконання всіх перерахованих дій бажано застосовувати можливості автоматизації, які надає відповідна СУБД. Для створення дружнього інтерфейсу користувача (у вигляді форми з елементами автоматизації дій), в СУБД Access можна використовувати наступні елементи керування: поля зі списками, групи перемикачів, різноманітні об’єкти ActiveX, новостворені опції головного та контекстного меню.

## 7.1. Поле зі списком

Під час введення даних формах бази даних Access можна швидко та легко вибрати значення зі списку, ніж запам'ятовувати значення для введення. Список варіантів також дає змогу введення достовірних значень. Елемент керування "список" може підключатися до наявного поля даних або відображати постійні значення, введені під час створення елемента керування [3].

### 7.1.1. Створення поля, значення якого вибираються із списку

Для створення поля зі списком за допомогою майстра необхідно відкрити форму в режимі конструктора, обрати піктограму **Поле зі списком**  та розмістити його на формі. Після чого запуститься майстер поля зі списком (рис. 7.1). Переконайтеся, що на вкладці **Конструктор** у групі Елементи керування активовано кнопку **Использовать мастера**  .

Далі потрібно обирати варіанти із запропонованих майстром: вибрати таблицю та поле джерела даних звідки будуть отримані значення списку та

дотримуватися вказівок, щоб вказати, як відображатимуться значення. Після вибору необхідних параметрів натиснути кнопку **Готово**. Тепер для кожного запису можна вибирати значення списку відповідного поля у формі.

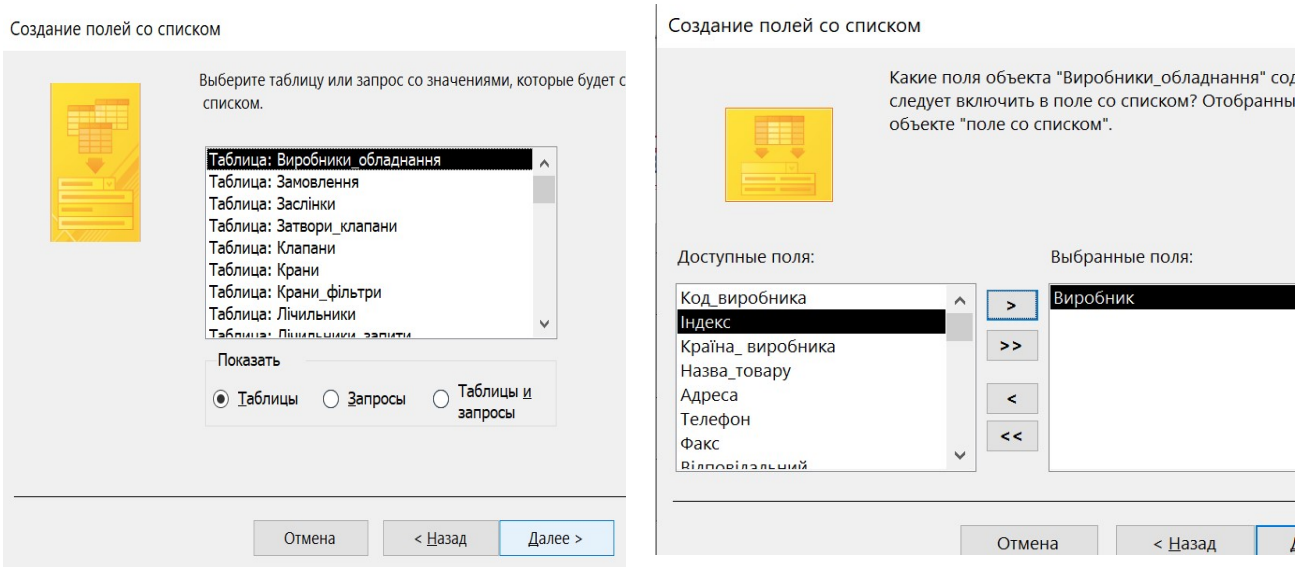



Рис. 7.1. Створення поля зі списком за допомогою майстра

Таким чином, поле **Виробник** таблиці **Виробники_обладнання** включатиме список назв всіх компаній виробників обладнання, назви яких вже існують в базі даних. Цей список буде доповнюватися назвами нових компаній виробників, якщо такі з’являтимуться.

### 7.1.2. Перетворення звичайного поля у поле, значення якого вибираються із списку

Поле із списком можна створити без майстра. Наприклад, перетворити попередньо створене на формі поле **Приєднання** у поле із списком.

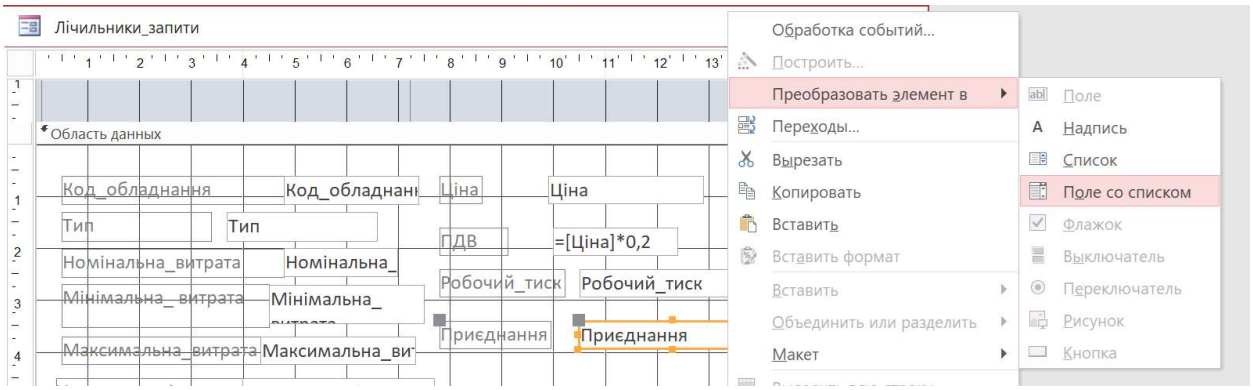



Рис. 7.2. Перетворення звичайного поля у поле зі списком за допомогою опції контекстного меню

Для чого відкрити форму у режимі конструктора та виділити відповідне поле. У контекстному меню вибрати опції **Преобразовать элемент** в **Поле со списком** (рис. 7.2). Потім у вікні **Окно свойств** відповідного поля в якості **Источника строк** обрати попередньо створений запит. Для наведеного приклада було створено запит з ім’ям Запит_приєднання (рис. 7.3). Або в рядку **Источник строк** безпосередньо написати наступну команду SQL.

***SELECT DISTINCT Лічильники_запити.Приєднання***

***FROM Лічильники_запити;***

Зарезервоване слово ***DISTINCT*** використовується, щоб запобігти повторенню значень у списку.

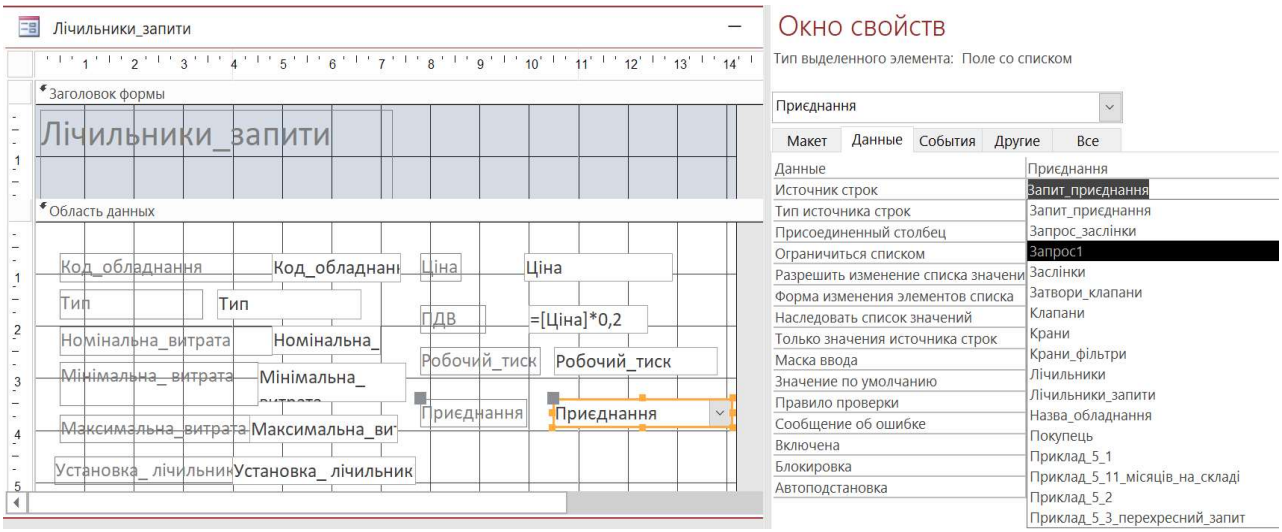



Рис. 7.3. Вибір джерела даних для поля зі списком **Приєднання**

На рис. 7.4 представлена форма з розгорнутим списком поля **Приєднання**. Перераховані у списку значення можна використовувати при введенні даних про новий лічильник або при редагування вже існуючих даних. Якщо при введенні нових даних з’являтимуться нові види приєднання, то ці дані будуть автоматично додаватися до існуючого списку. Це можливо тому що у властивостях поля обрано значення **Да** для **параметра Разрешить изменение списка значений.**

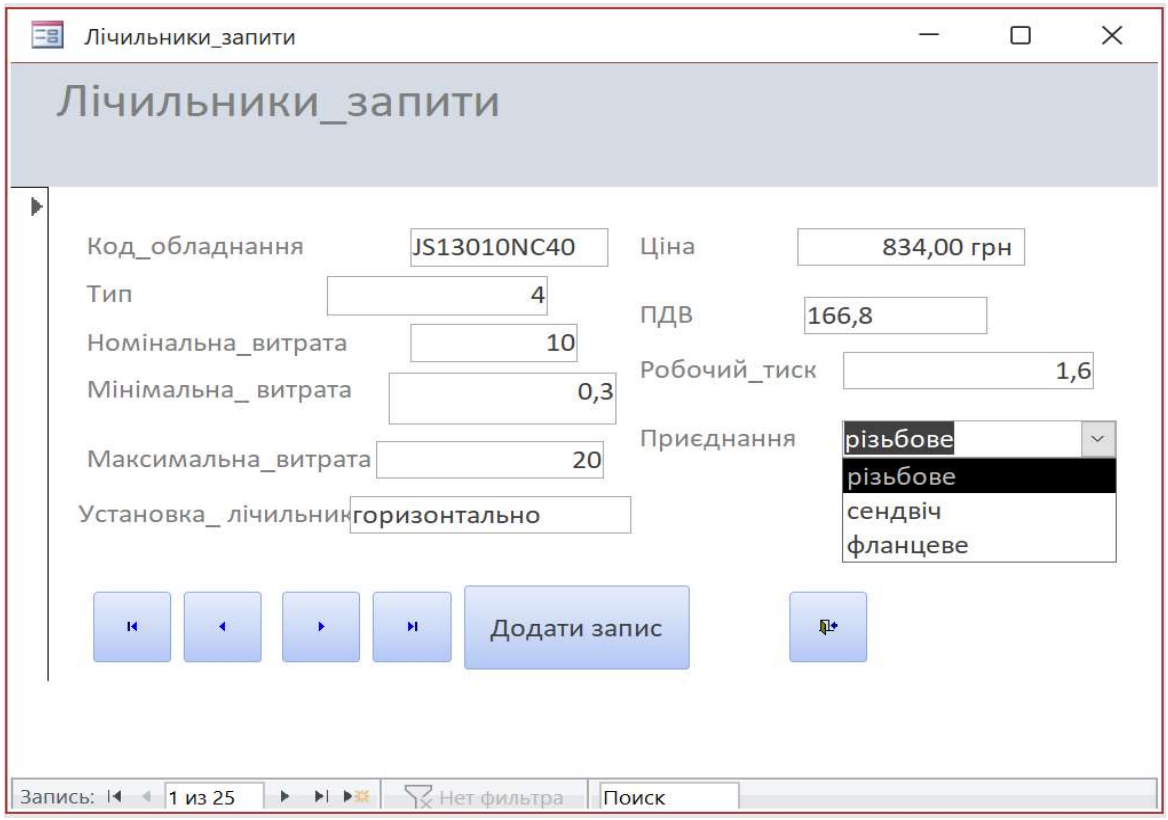



Рис. 7.4. Форма **Лічильники_запити** з модифікованим у поле зі списком поля даних **Приєднання**

## 7.2. Елемент управління Список

Елемент керування **Список** відображає список значень або варіантів. Список містить рядки даних, і зазвичай цей список складається з декількох рядків, які мають бути видимими в будь-який час. Рядки можуть мати один або декілька стовпців, які можуть відображатися з заголовками або без них. Якщо список містить більше рядків, ніж можна відобразити в елементі керування, у програмі Access буде відображатися смуга прокручування в цьому елементі керування. Створений список обмежений переліком варіантів, наведених у списку. Користувач не може вводити нові значення до списку або редагувати існуючі значення**.**

### 7.2.1. Запуск майстра створення список

Для створення елемента керування **Список** при використанні програми Майстра піктограма **Использовать мастера** на панелі елементів має бути

активована. Після вибору на панелі елемента управління  **Список**, необхідно намалювати контури елемента на формі. При цьому програма автоматично відкриє відповідну програму М**айстер**.

Значення звичайного або комбінованого списку беруться з поля таблиці або запиту. Приміром, при створенні форми **Замовлення** список виробників обладнання краще оформити у вигляді звичайного списку, при цьому наповнивши його значеннями з поля **Виробник** таблиці **Виробники обладнання**. Наступна послідовність дій продемонструє роботу з майстром списків для варіанта наповнення його з поля запиту або таблиці. У першому діалоговому вікні **Майстра** виділити перемикач **Об'єкт буде використовувати значення з таблиці або запиту** (рис. 7.5)**.** Клацніть на кнопці **Далее.**

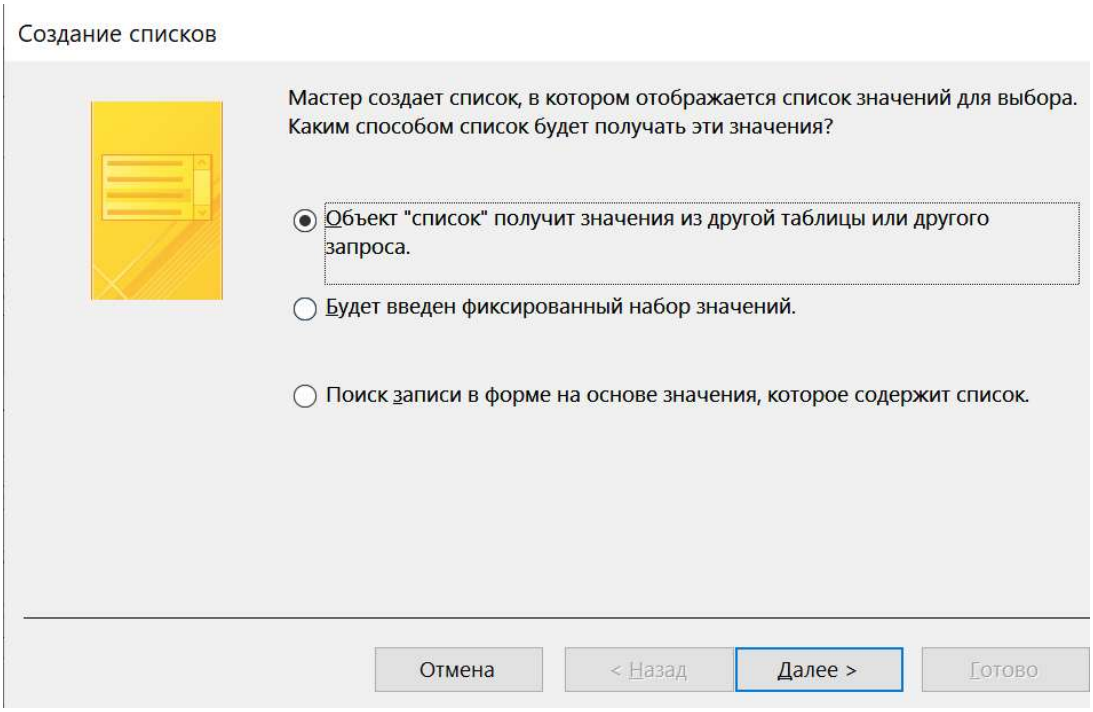



Рис. 7.5. Вікно для вибору варіанта використання даних для елемента керування **Список**

Коли майстер запитає, як потрібно отримати значення для елемента керування, необхідно обрати одну з наведених нижче дій (рис. 7.5).

- Якщо потрібно відобразити поточні дані з джерела записів, натисніть кнопку **«Объект «список» получит значения из другой таблицы или другого запроса»**.

- Якщо потрібно відобразити зафіксований список значень, який рідко змінюватиметься натисніть кнопку **«Будет введен фиксированный набор значений»**.
- Якщо потрібно, щоб елемент керування виконував операцію пошуку, а не використовувався в якості засобу введення даних, натисніть кнопку **«Поиск записи в форме на основе значения, которое содержит список»**. Після цього створюється вільний елемент керування з вбудованим макросом, який виконує операцію пошуку на основі значення, яке вводить користувач.

За умови вибору першого варіанту відкривається вікно для джерела даних _ таблиці чи запиту.

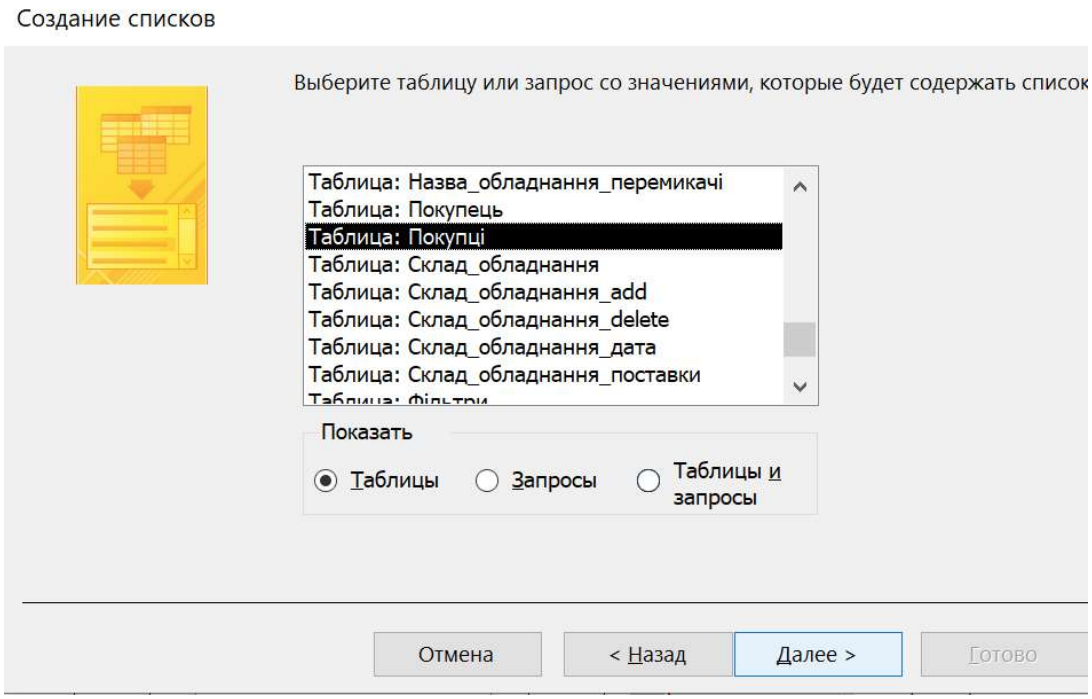



Рис. 7.6. Вікно для вибору джерела даних для елемента керування **Список**

Якщо ви вибрали один із двох перших параметрів на першій сторінці майстра, майстер запитає, що потрібно зробити, коли ви виберете значення. Маємо виконати одну з таких дій:

- Щоб створити вільний елемент керування, натисніть кнопку запам'ятати значення для подальшого використання. Це означає, що застосування використовуватиме вибране значення, доки користувач не змінить його або не закриває форму. Але не це значення до таблиці не буде записуватися.

- Щоб створити зв’язаний елемент керування, необхідно вибрати поле з попередньо обраного джерела даних, в якому буде зберігатися значення списку (рис. 7.7).

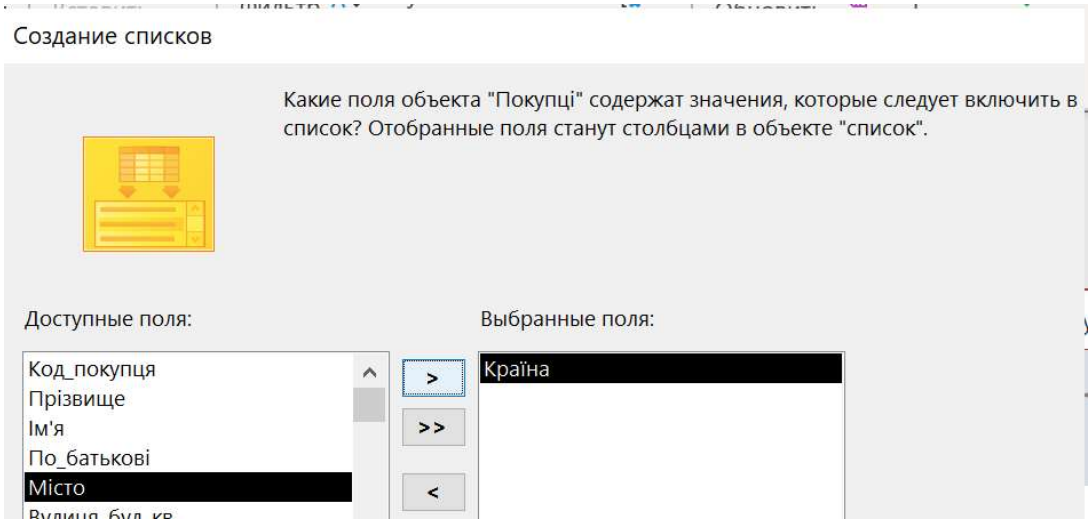



Рис. 7.7. Вікно для вибору поля джерела даних для збереження значень об’єкта **Список**

- Натиснути кнопку **Далее** та введіть підпис для елемента керування. Цей підпис відображатиметься поруч із елементом керування.

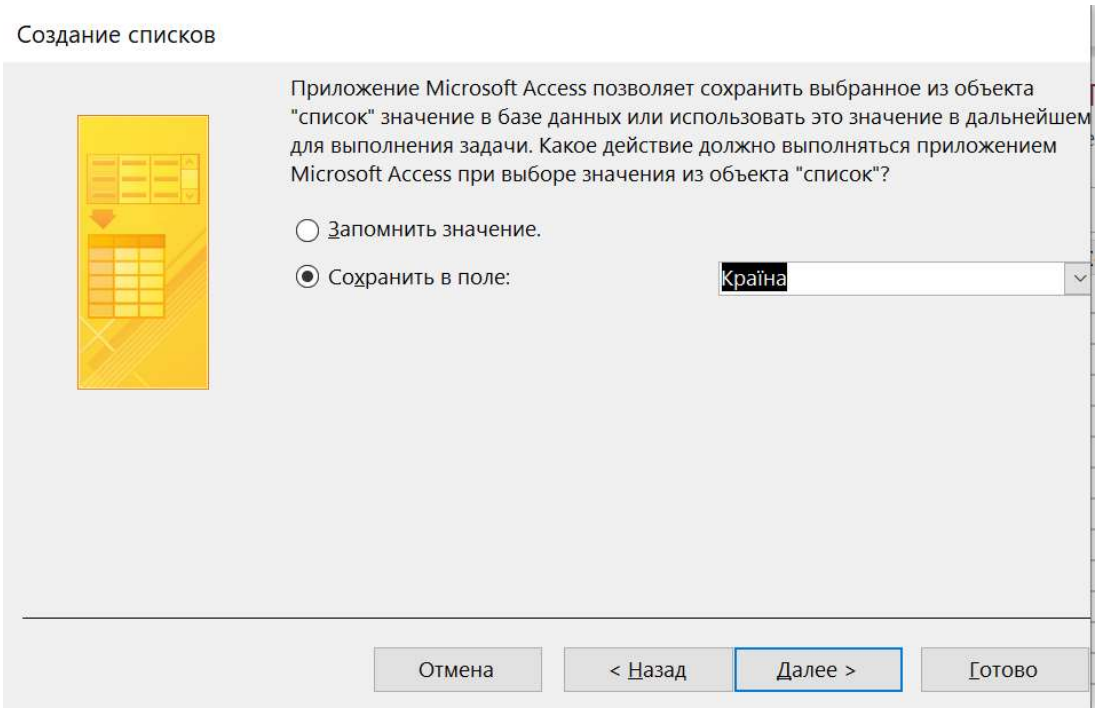



Рис. 7.8. Вікно для вибору використання значень об’єкта **Список**

Завершується створення елементу керування Список наданням йому смислового варіанту імені.

- Натиснути кнопку **Готово**.

В даному варіанті створеного списку, обране значення списку буде запам’ятовуватися у відповідному полі. Зверніть увагу на той факт, що при створенні списку було запропоновано включити в список і ключове поле, але не виводити його в список ( у властивостях списку цьому стовпчику задано ширину 0 см). Тому у властивостях створено списку потрібно обрати для роботи із значенням Країна другий стовпчик (для властивості списку **Присоединенный столбец** – обрати значення 2).

Якщо використовувати форму для перегляду даних, то виконуючи перехід від одного запису до іншого в полі **Список** виділеним буде значення, яке вже в цьому полі відповідної таблиці запам’ятовано. Значення поля можна змінити обравши нове значення із списку. Також значення списку можна використовувати для введення нових даних, наприклад, про нового покупця.

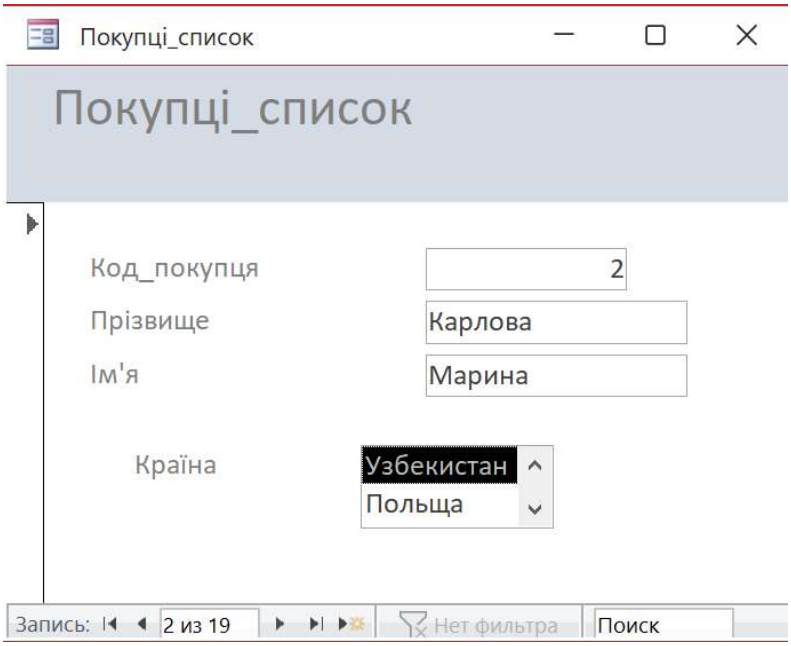



Рис. 7.9. Вигляд форми з використаним об’єктом **Список** для поля **Країна**

### 7.2.2. Ручне перетворення поля у поле Список

Для того щоб перетворити попередньо створене поле у поле **Список** потрібно відкрити форму у конструкторі та виділити відповідне поле. У контекстному меню вибрати опції **Преобразовать элемент** в **Список.** Для властивості **Источник строк** обираємо таблицю( в нашому прикладі таблицю Покупці). Для перетворення у Список обрано поле Ім’я. Це поле у таблиці

знаходиться у третьому стовпчику, тому для властивості **Присоединенный столбец** задаємо значення 3 (рис. 7.10).

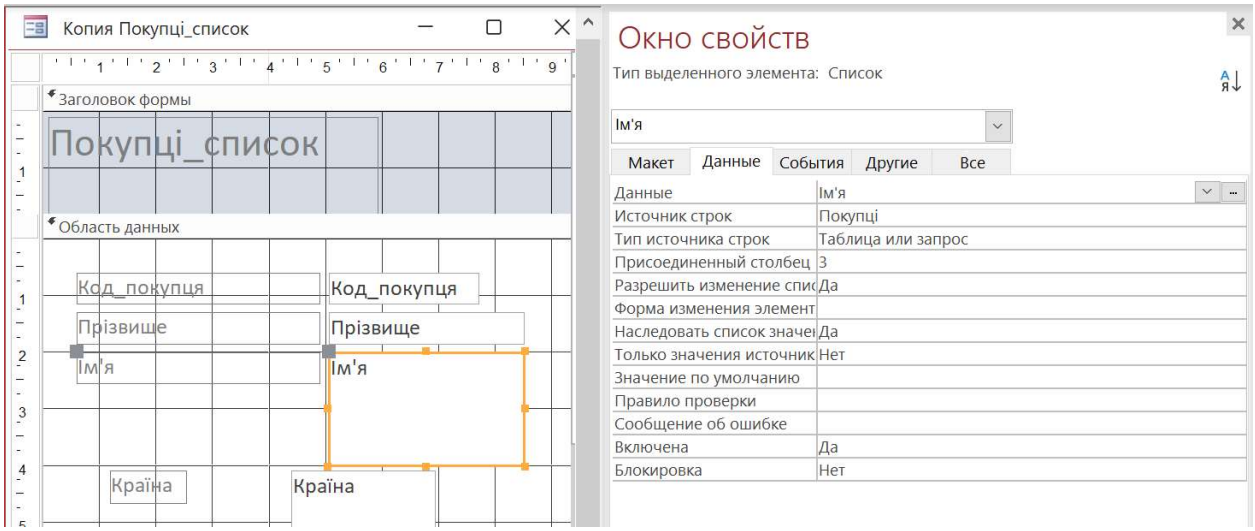



Рис. 7.10. Вигляд форми в режимі конструктора з властивостями **Источник строк** та **Присоединенный столбец** об’єкта **Список** для поля **Ім’я**

Але щоб значення двох попередніх стовпчиків у список не виводилися, у властивості **Ширина столбцов** задаємо їм нульову ширину (рис. 7.11).

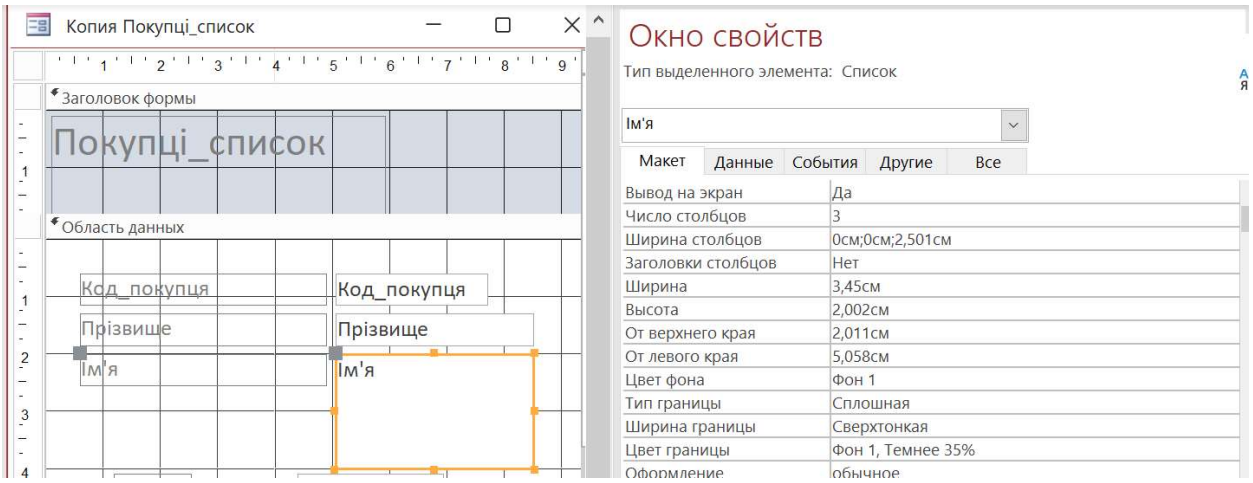



Рис. 7.11. Вигляд форми в режимі конструктора з властивостями **Число стобцов** і **Ширина столбцов** об’єкта **Список** для поля Ім’я

Таким чином, переглядати дані про ім’я покупця чи вводити ім’я нового покупця можна використовуючи список.

## 7.3. Група перемикачів

Перемикачі призначені для того, щоб у ході виконання об’єкта бази даних вибрати один з можливих варіантів.

На вкладці Конструктор у групі елементи керування розташовані елементи керування – перемикачі: прапорець , радіокнопка , кнопка перемикач .

Прапорці призначені для того, щоб у ході виконання проекту вибрати або один з можливих варіантів, або декілька. Наприклад, установити значення однієї або кількох властивостей об’єкта з деякого визначеного набору властивостей.

З групи перемикачів може бути вибраний тільки один, а в групі прапорців позначка може бути встановлена або на одному з них, або на декількох, або на жодному.

Наприклад, установити для певної властивості об’єкта одне значення з деякого визначеного набору значень.

### 7.3.1. Створення групи перемикачів за допомогою майстра

Розглянемо порядок створення на формі елемента керування **Группа переключателей**. Для цього потрібно натиснути на піктограму 

на панелі інструментів та виділити область на формі, у якій має бути розташована група перемикачів. Далі відкриється діалогове вікно «Создание группы переключателей». У колонці Подписи вводимо значення даних поля, для якого створюється група перемикачів, наприклад, лічильник ВЄПС для назви обладнання або числове значення 1,5 для похибки вимірювання. Вигляд діалогового вікна «Создание группы переключателей» продемонстровано на рис. 7.12.

Після цього за допомогою кнопки **Далее** переходимо до наступного кроку. У вікні, що відкрилося надається можливість встановити значення перемикача, яке буде записане у відповідне поле за замовчуванням. Тобто в прикладі (рис. 7.13) обрано 1 в якості значення за замовчуванням.

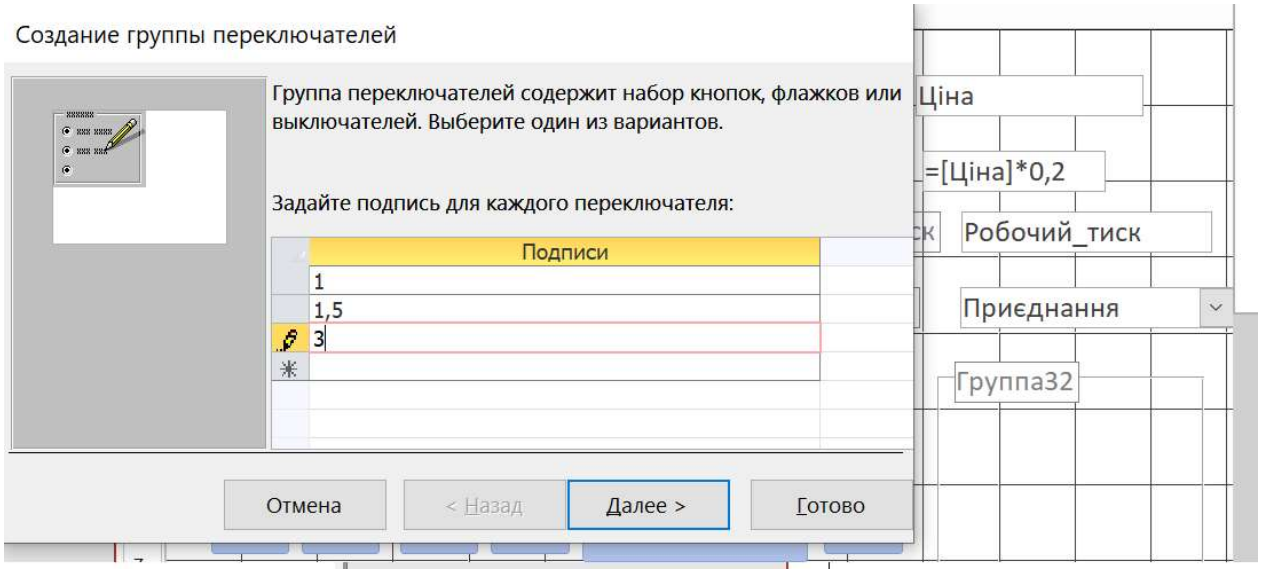



Рис. 7.12. Вікно «Создание группы переключателей» для оформлення підписів

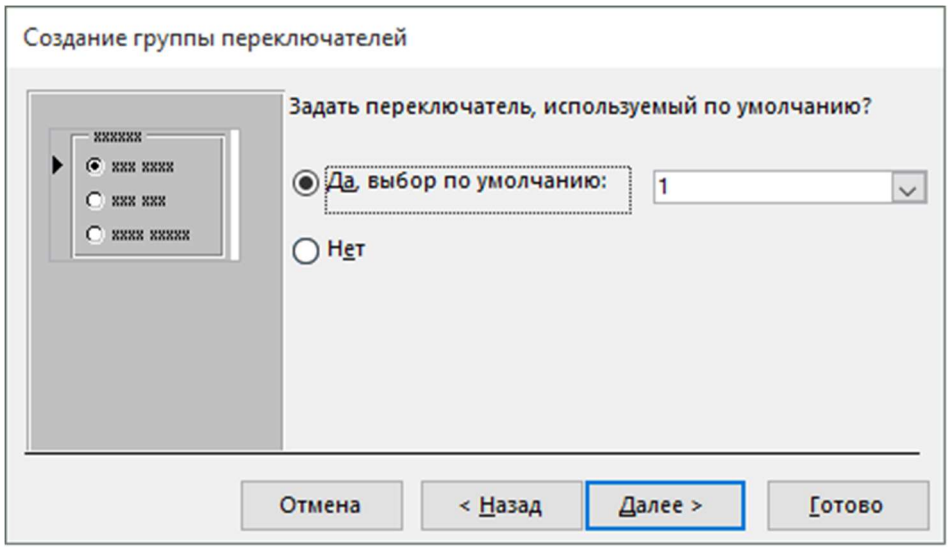



Рис. 7.13. Вибір значення за замовчуванням

Не змінюючи значень вікна натискаємо кнопку **Далее**. Після чого з'явиться вікно для встановлення значень перемикачам групи. На цьому етапі у колонку **Значения** вводимо відповідні значення , що будуть заноситися до поля **Похибка вимірювання** при виборі відповідного елемента керування групи. В нашому прикладі підписи і значення співпадають (рис. 7.14).

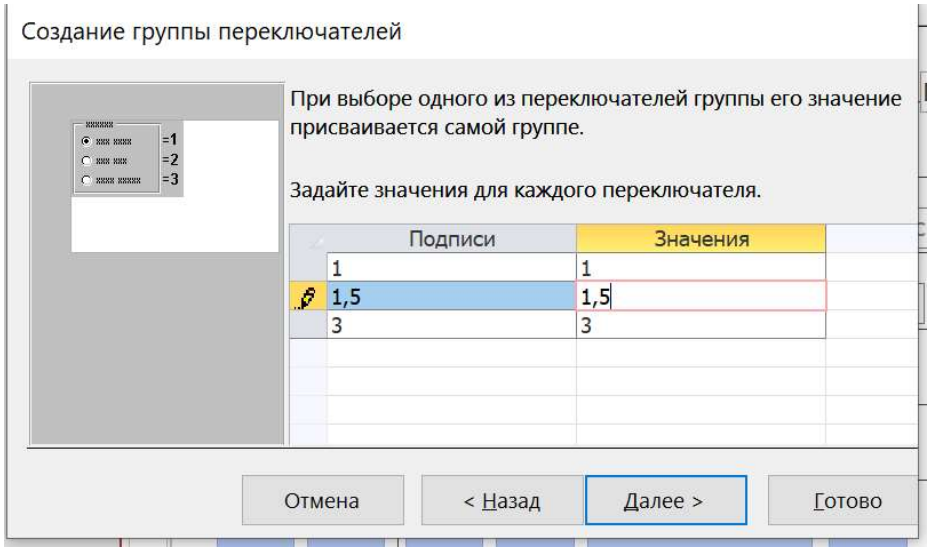



Рис. 7.14. Вікно для встановлення відповідних значень перемикачам групи

Обираємо **Сохранить значение в поле** та вказуємо відповідне поле - **Похибка вимірювання** (для нашого приклада).

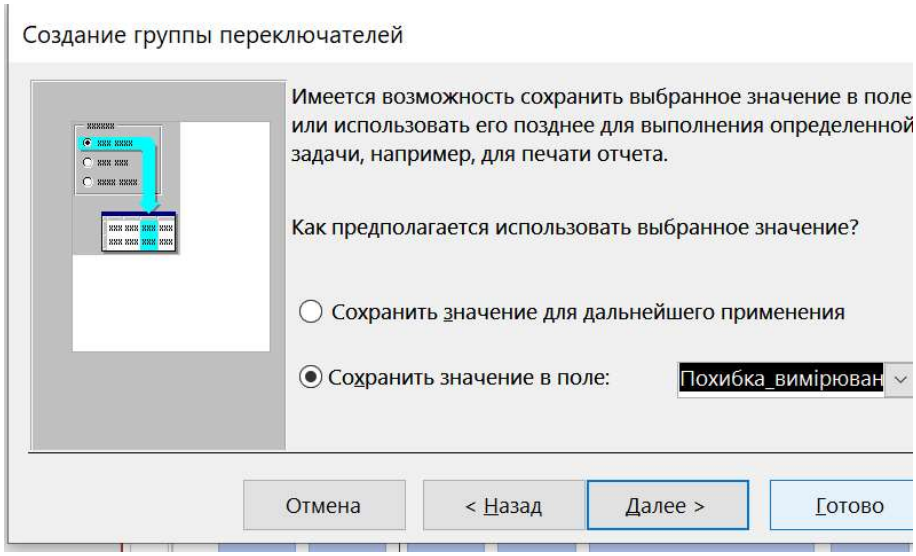



Рис. 7.15. Вікно вибору поля таблиці для збереження обраного значення перемикача

В полі значення перемикача вказуємо назву перемикача. Виконуємо подібні дії з кожним із перемикачів та натискаємо кнопку **Далее**. З'являється наступний крок налагодження групи перемикачів, в якому вказуємо поле **Похибка вимірювання** (ця змінна вказує на те, що значення групи перемикачів буде запам'ятовуватись у вказаному полі). Необхідно звернути увагу на той факт, що в обране поле буде вставлятися не задане нами значення перемикача, а порядковий номер цього значення в списку перемикачів. На наступному кроці можемо обрати тип відображення перемикачів (рис. 7.16)

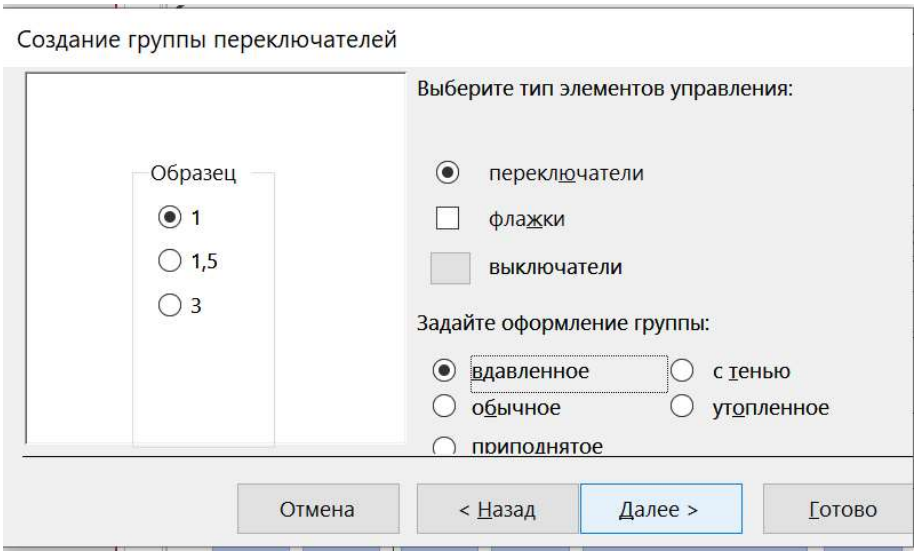



Рис. 7.16. Вікно для вибору зовнішнього вигляду елемента керування

Натискаємо кнопку **Далее**. У наступному кроці здійснюється встановлення типу групи. Вводимо назву для групи перемикачів та натискаємо **Готово.**

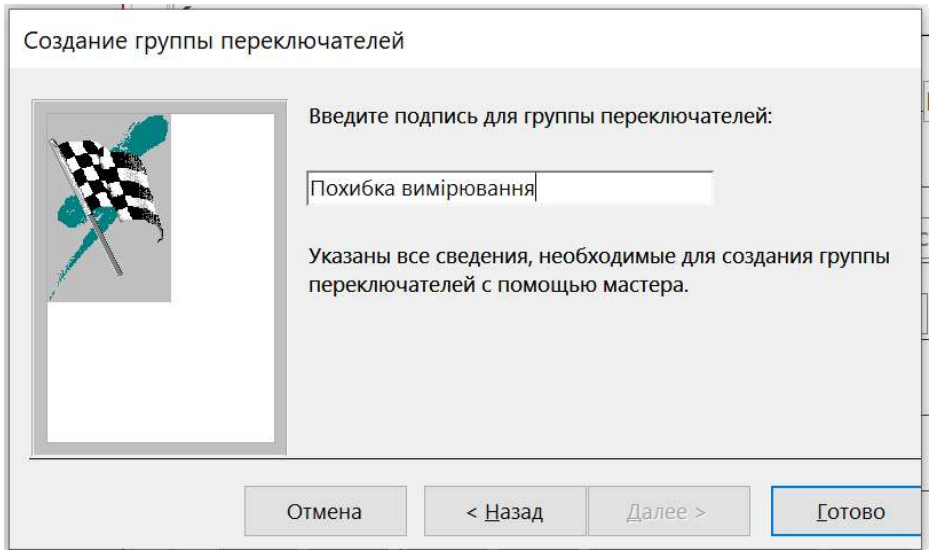



Рис. 7.17. Вікно для введення назви елемента керування **Группа переключателей**

В результаті отримаємо форму зі створеною групою перемикачів (рис. 7.18).

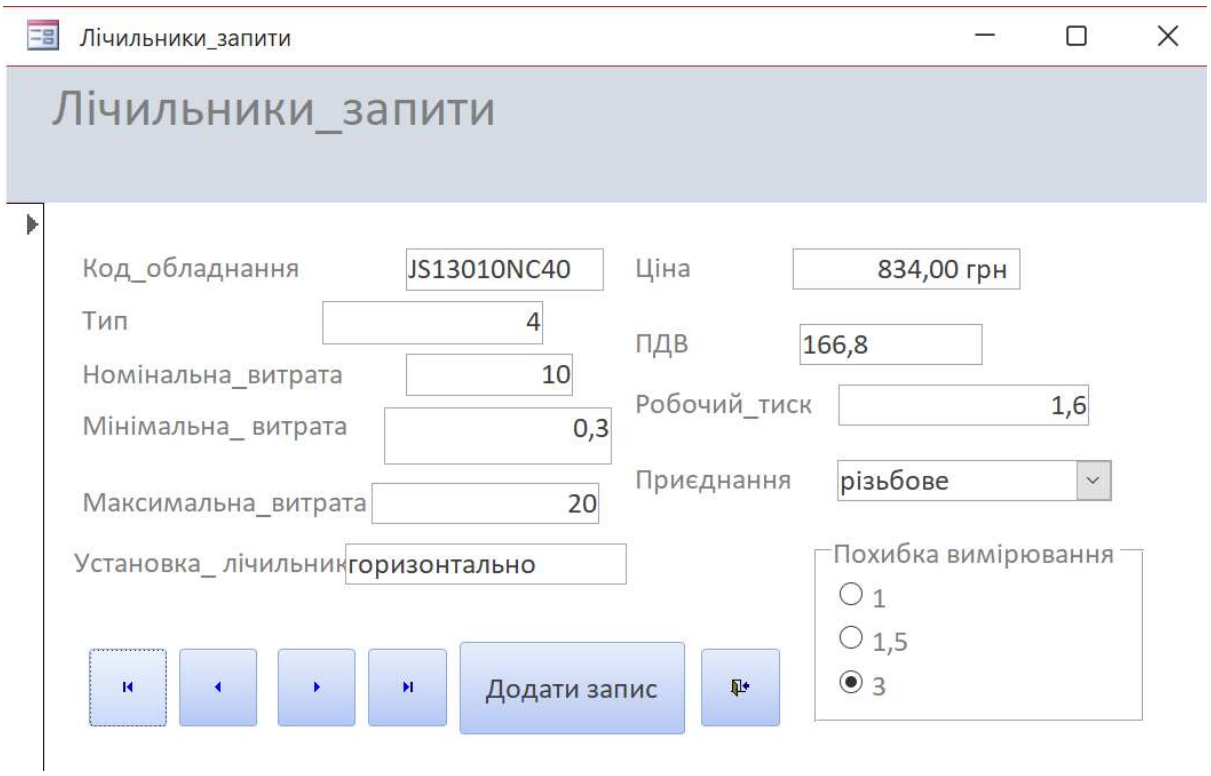
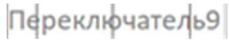


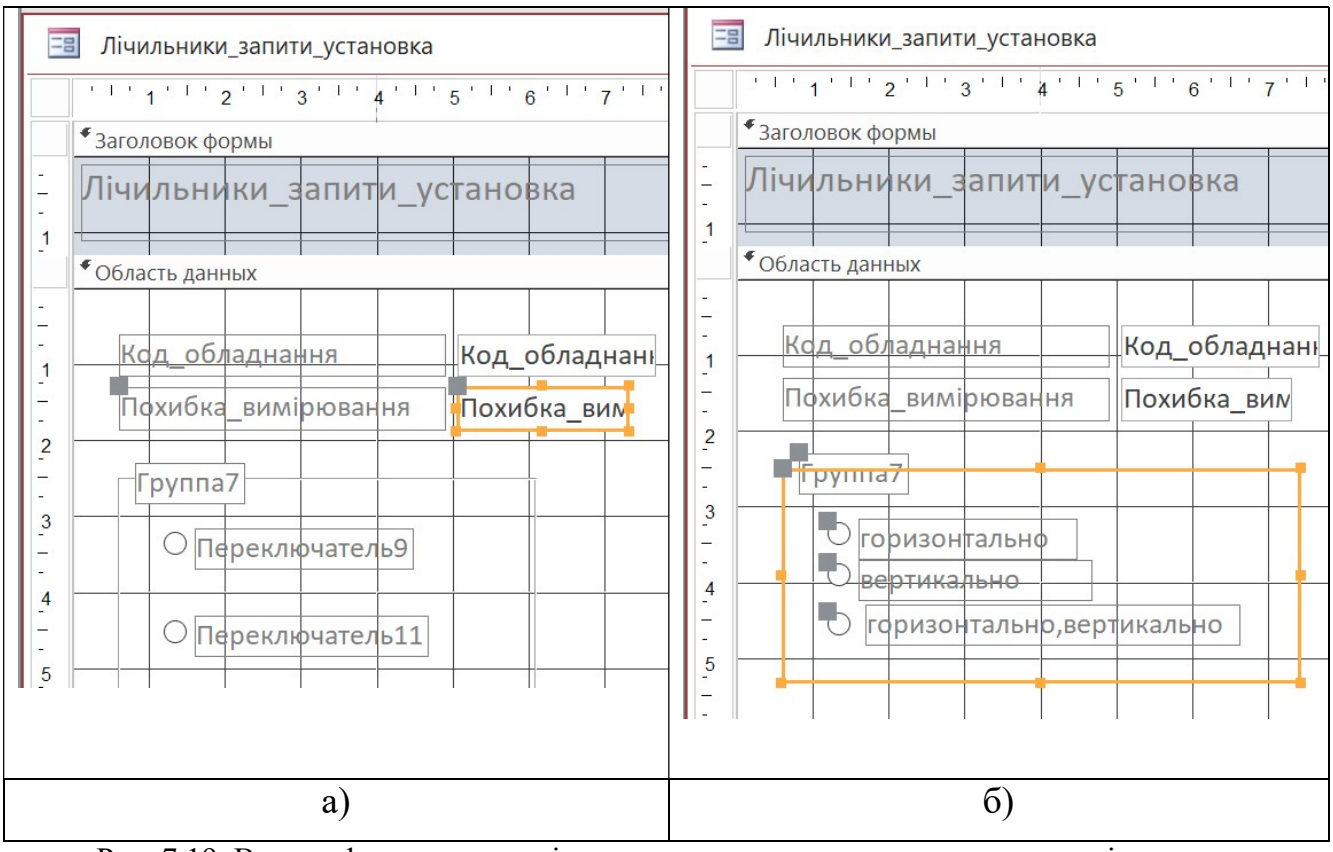

Рис. 7.18. Вигляд форми з групою перемикачів

Створена група перемикачів може використовуватися як для перегляду даних так і введення нових даних у поле **Похибка вимірювання** відповідної таблиці.

### 7.3.2. Створення групи перемикачів вручну

Для того щоб створити групу перемикачів вручну, необхідно відключити кнопку **Использовать мастера**  та виділивши піктограму **Група перемикачів**, окреслити на формі контури групи перемикачів. Далі обрати Перемикач на панелі елементів, вставити його всередині контуру групи. Кількість перемикачів і їх зовнішній вигляд вибираються відповідно до завдання.

Кожен перемикач складається з елемента керування  та підпису до нього  . При виборі опції контекстного меню **Свойства** для виділеного елемента керування, у вікні, що відкриється, обираємо необхідні параметри для налаштування наших елементів керування почергово.




а) б)

Рис. 7.19. Вигляд форми в режимі конструктора для створення перемикачів вручну

Так на вкладці **Все** у рядку **Подпись** для кожного елемента **Надпись** необхідно ввести текст, що повинен відображатися поруч із перемикачем (рис. 7.20). А підпис **Группа7** замінити на **Установка лічильників**.

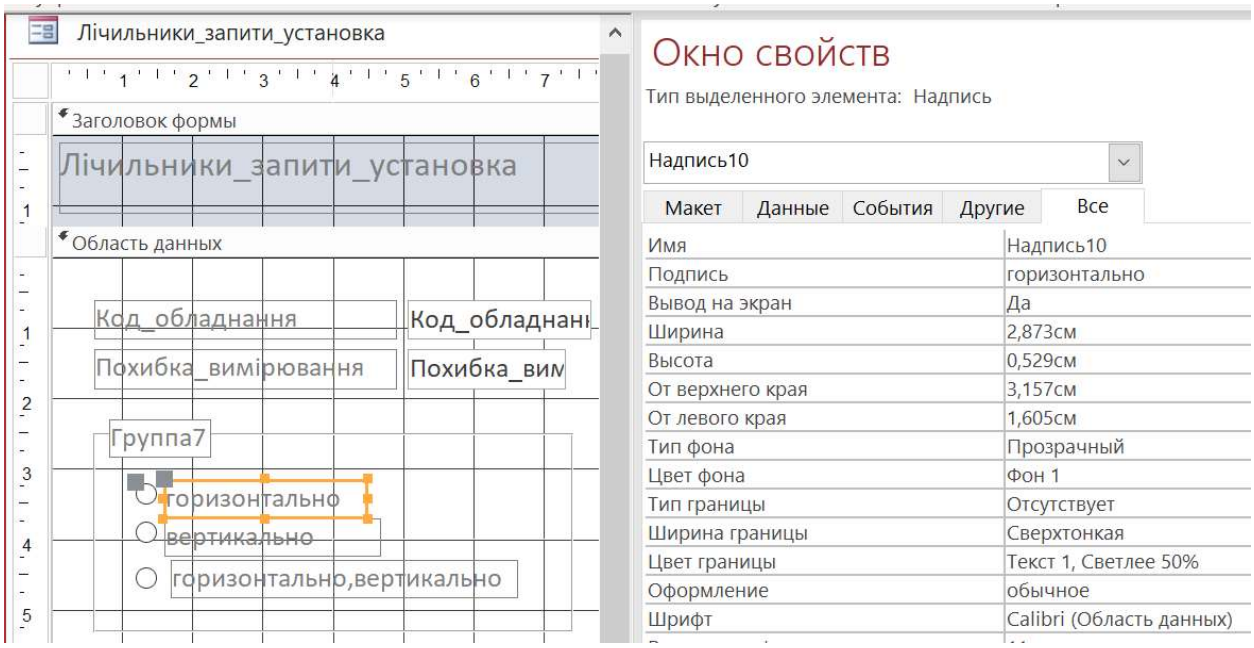



Рис. 7.20. Вигляд форми в режимі конструктора з властивостями **Надписи**

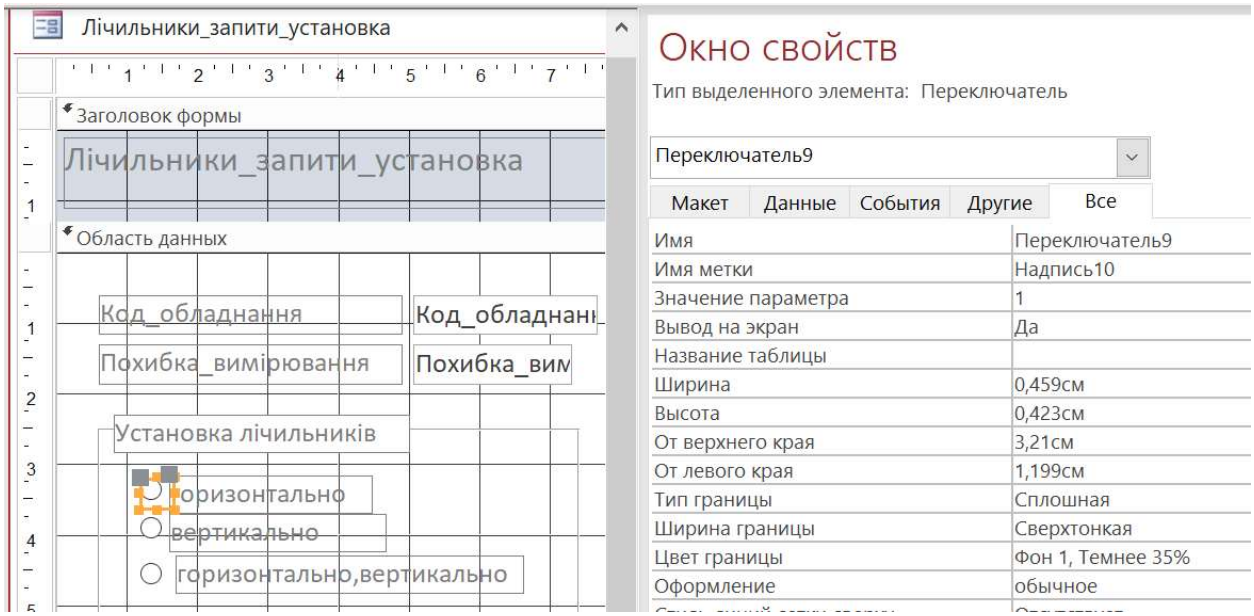



Рис. 7.21. Вигляд форми в режимі конструктора з властивостями **Переключателя**

На вкладці **Все** в рядку **Значение параметра** введіть числове значення, асоційоване з положенням перемикача. Номер назви об’єкту (в нашому прикладі назви радіокнопки) буде на одиницю більше номера відповідного елемента перемикача. Приміром, якщо ім’я радіокнопки, яке було автоматично привласнено програмою Access – **Переключатель9**, то відповідний напис буде мати назву **Надпись10** (рис. 7.21). Необхідно повторити описані вище дії для всіх перемикачів Групи.

На вкладці **Данные** в рядку **Данные** необхідно вибрати із списку поле таблиці, що буде асоційовано із створеною групою перемикачів (рис. 7.22).

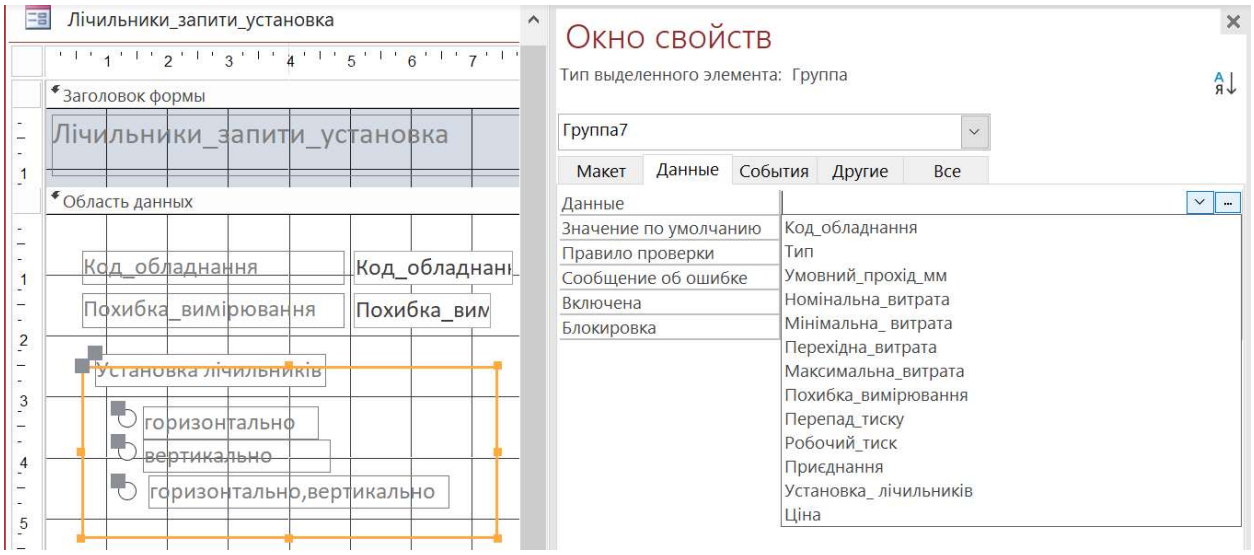



Рис. 7.22. Встановлення зв’язку елементів керування групи з відповідним полем

Якщо потрібно, щоб при створенні нового запису в поле постійно вводилося одне і теж саме значення, його потрібно ввести в рядок **Значення за замовчуванням**. Після цього вікно **Окно Свойств** можна закрити.

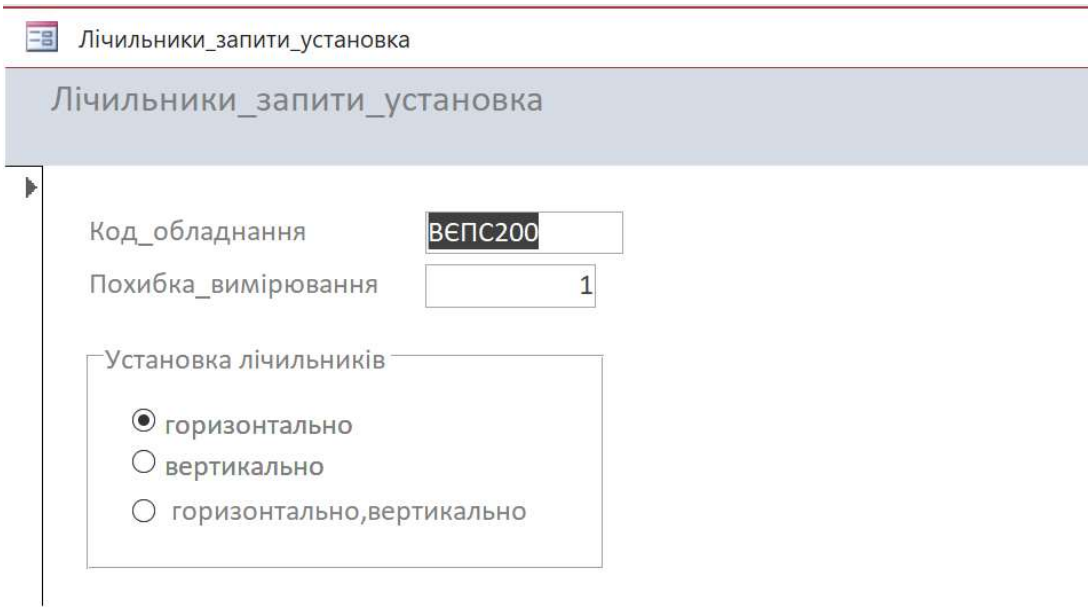


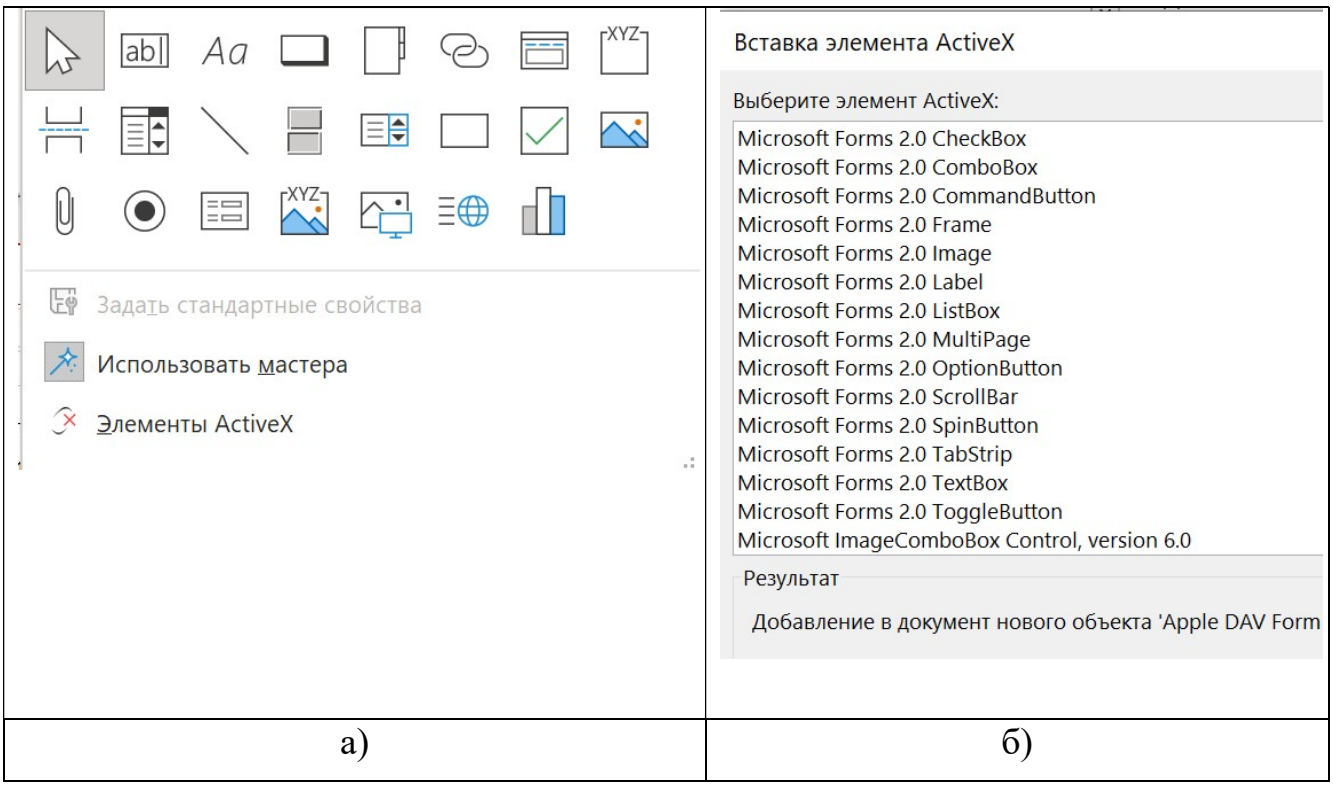

Рис. 7.15 Вигляд форми з групою перемикачів створених вручну

Як і в попередньому прикладі, створена група перемикачів може використовуватися як для перегляду даних так і введення нових даних у поле Установка лічильників відповідної таблиці.

## 7.4. Введення даних за допомогою елементів ActiveX

Керуючі елементи ActiveX — це як будівельні блоки для програм, вони можуть бути використані для спрощення введення даних у формах. В MSAccess можна обрати необхідний елемент ActiveX зі списку (рис. 7.16,б).

Для розгортання списку з переліком елементів керування ActiveX, які можна вставляти в форми, необхідно вибрати на панелі елементів керування опцію **Элементы ActiveX** (рис. 7.16,а)**.** Потім виділити знайдений у списку елемент, та перетягти його на форму.




а) б)

Рис. 7.16. Вибір елемента керування ActiveX

### 7.4.1. Введення чисел за допомогою елемента ActiveX (SpinButton)

Введення чисел можна організувати за допомогою елемента **ActiveX** . Елемент **ActiveX** має назву - Microsoft Forms 2.0 SpinButton, вибравши потрібний елемент **ActiveX**, малюємо його контур на формі і отримуємо кнопку. Access надає цьому об’єктові ім’я, в нашому прикладі **SpinButton9.**

Використаємо Microsoft Forms 2.0 SpinButton для введення значень у поле Номер_замовлення форми Замовлення_нова. Для цього клікаємо мишею на імені об’єкта Microsoft Forms 2.0 SpinButton і перетягуємо отриману кнопку до поля Номер_замовлення.

На формі з’явився новий об’єкт **SpinButton9**. Виділяємо об’єкт **SpinButton9** та у вікні його властивостей у полі Данные обираємо поле Номер_замовлення. Тобто зв’язуємо об’єкт **SpinButton** з полем, до якого будуть вводитися дані з використанням **SpinButton9**.

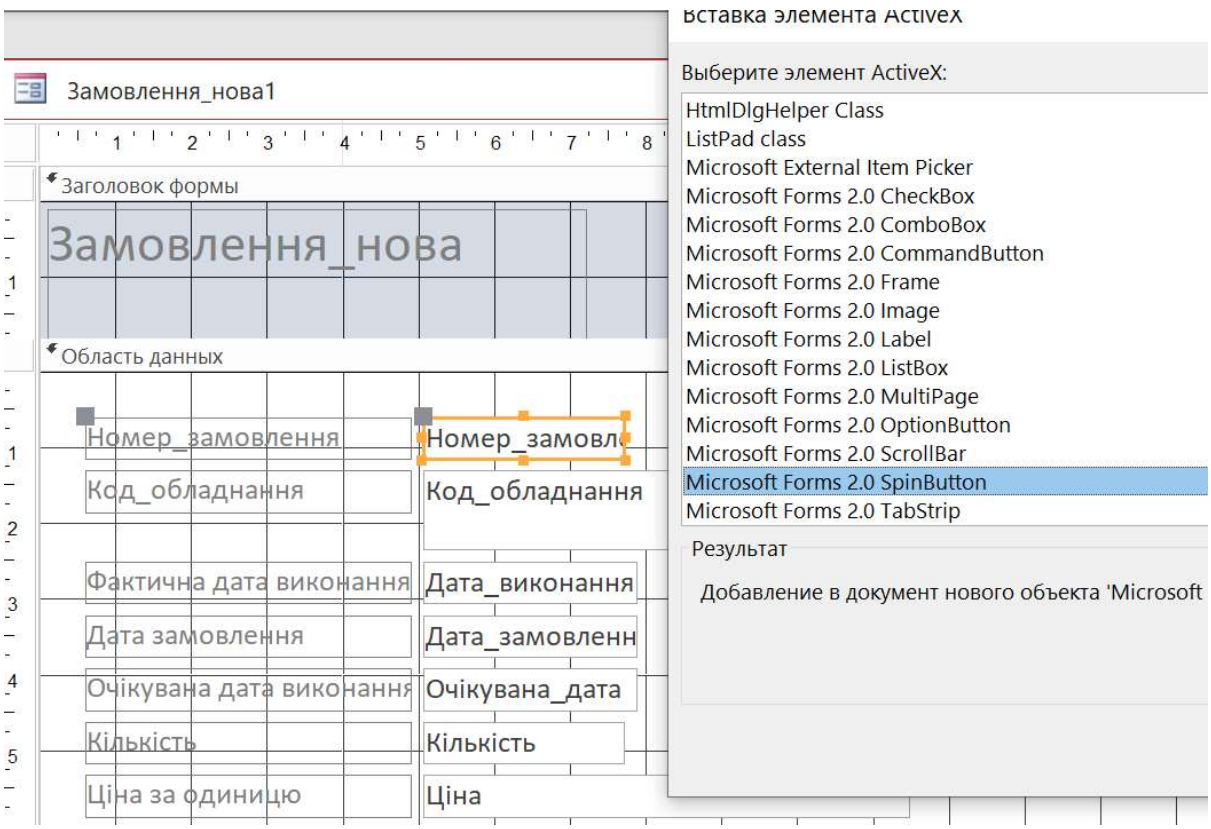



Рис. 7.17. Вибір зі списку об’єкта Microsoft Forms 2.0 SpinButton

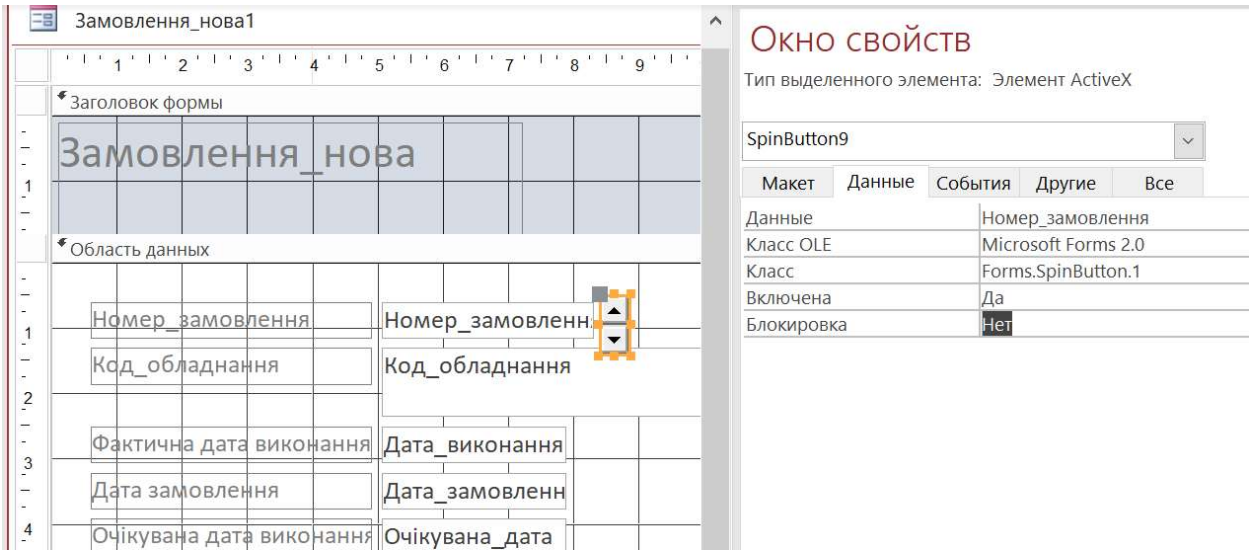



Рис. 7.18. Вибір у вікні властивостей поля, до якого будуть вводитися дані з використанням **SpinButton9**

Далі у вкладці **События** у рядку події **При обновлении** натискаємо на піктограмку і у вікні що відкриється виберемо Программы.

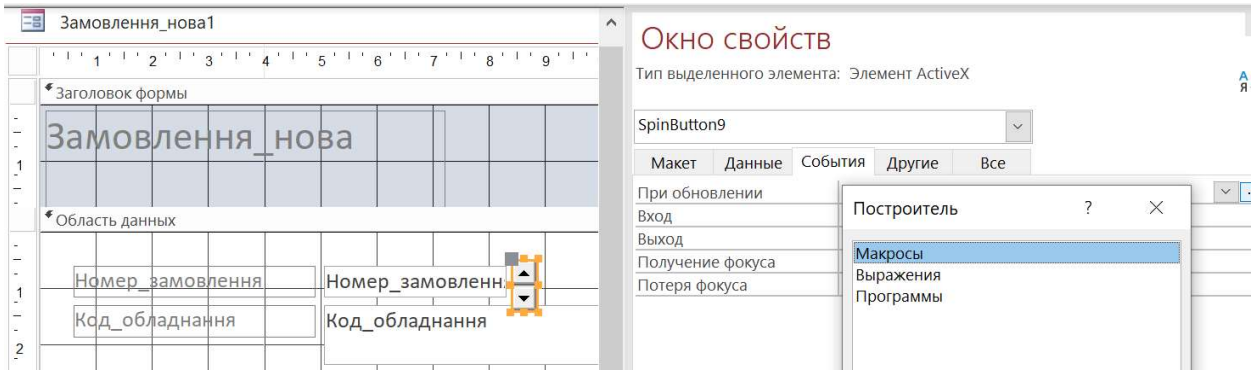



Рис. 7.19. Виклик редактора Visual Basic для програмування події **При обновлении**

Відкриється вікно редактора Visual Basic, в якому потрібно набрати текст наступної процедури.

***Private Sub SpinButton9_Updated(Code As Integer)***

***Номер_замовлення.SetFocus***

***End Sub***

В результаті отримаємо форму з новим полем Номер_замовлення яке дані до якого вводяться за допомогою **SpinButton9.**

Для коректного введення номера телефону різними користувачами, обов'язково необхідно створити маску вводу для цього поля. Шаблон маски прописується у рядку **Маска ввода** властивостей поля **Телефон**.

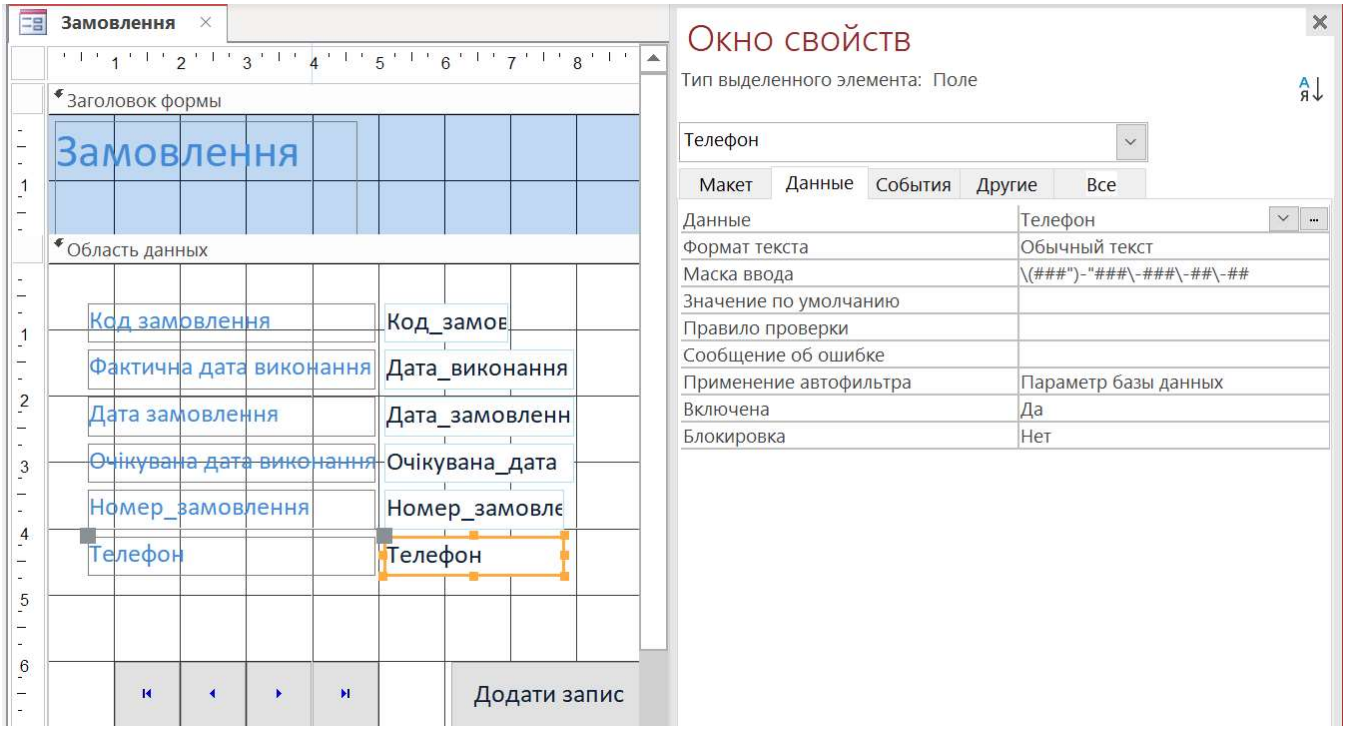



Рис. 7.20. Шаблон маски ввода поля **Телефон**

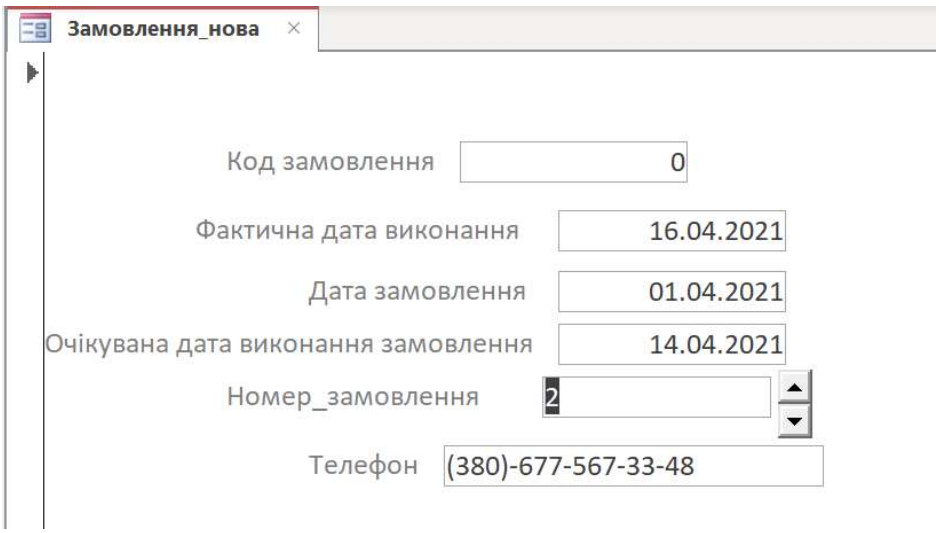



Рис. 7.21. Вигляд форми з елементом ActiveX **SpinButton9 та маскою вводу номера телефон**

### 7.4.2. Введення дати за допомогою елемента ActiveX (Календарь)

Починаючи з версії MS Access 2013 Microsoft зробила даний інструмент окремим елементом в пакеті. В Microsoft Access 2013 вже включений елемент керування Calendar. оскільки спочатку додана підтримка елементів управління ActiveX. Тому при створенні форми, до полів типу Дата/время автоматично підключається елемент керування ActiveX Calendar. Якщо ж елемент керування Calendar не включений в використовувану версію MS Access, його необхідно доустановити, використовуючи інсталяційний диск MS Office.

## 7.5. Вибір стилю форми

При розробці форм слід дотримуватися таких правил:

- форма має бути простою. Бажано використовувати стандартні шрифти й кольори, які легко читаються та відповідають правилам ергономіки. Графіки та інші елементи, які підвищують інформативність форми, мають не перевантажувати форму;
- якщо в базі даних використовується декілька форм бажано використовувати один і той самий стиль для всіх форм;
- добираючи кольори для екранних форм, необхідно брати до уваги, що деякі монітори з низькою роздільною здатністю в графічному режимі або з поганим відтворенням кольорів не можуть правильно відобразити вашу

форму. Тому, щоб уникнути такої проблеми бажано використовувати установки для графічної карти й монітора найнижчого рівня;

- якщо в якості фонового зображення форми є необхідність використати фото, то необхідно пам’ятати, що такий об’єкт збільшує суттєво об’єм бази даних, тому такий стиль потрібно використовувати лише у разі крайньої потреби.

### 7.5.1 Вставка в форму фонового зображення

Для вставки в форму фонового зображення необхідно відкрити форму у режимі конструктора. Далі на вкладці **ФОРМАТ** обрати функцію **Фоновый рисунок.** Вибираючи фоновий рисунок, потрібно пам’ятати про відповідність вимогам ергономіки, тобто щоб створена форма була комфортною для роботи користувача. Якщо серед наявних рисунків відсутній рисунок, відповідний вимогам бази даних, то його потрібно створити і потім використати у якості фону.

### 7.5.2. Зміна стандартного стилю форми

Стиль форми можна легко змінювати за допомогою стандартних функцій : Темы, Цвета, Шрифты.

Обираючи піктограму **Темы**, можемо змінити як загальний вигляд форми, у тому числі і шрифти.

Обираючи піктограму **Цвета**, можемо швидко змінити всі кольори цієї форми на іншу палітру.

Обираючи піктограму **Шрифты**, можемо швидко змінити весь шрифт документа шляхом вибору нового набору шрифтів.

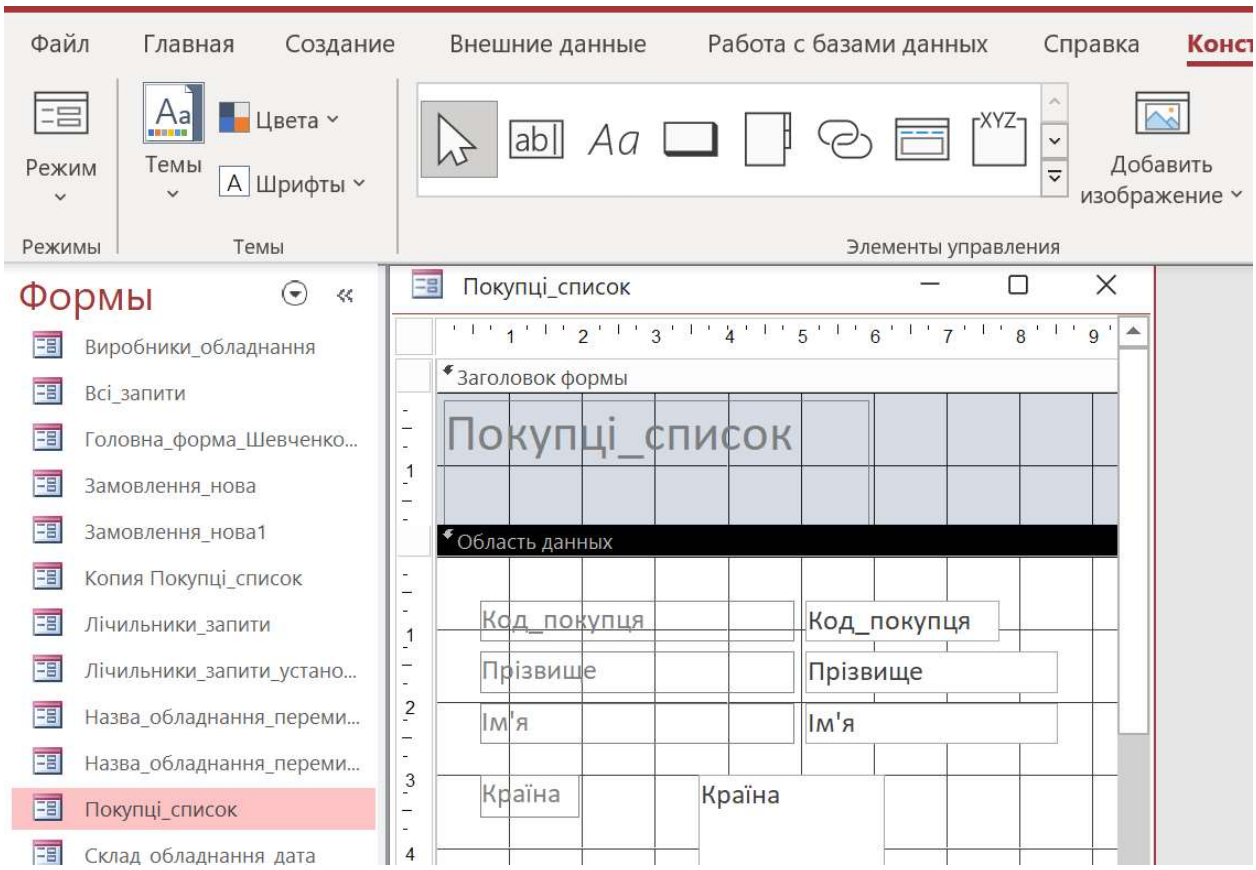



Рис. 7.22. Вибір стандартних функцій **Темы, Цвета, Шрифты** для зміни стилю форми

Обираючи будь-який варіант зміни стилю форми, потрібно пам’ятати про головне правило – форма повинна мати вигляд комфортний для подальшої роботи користувача.

## 7.6. Завдання до комп'ютерного практикуму

1. Відредагувати головну форму, створену на комп'ютерному практикумі №6. Вигляд форми (рис. 7.23). Програмувати подію натиснення кнопки будемо після виконання практикуму по створенню відповідних об’єктів.

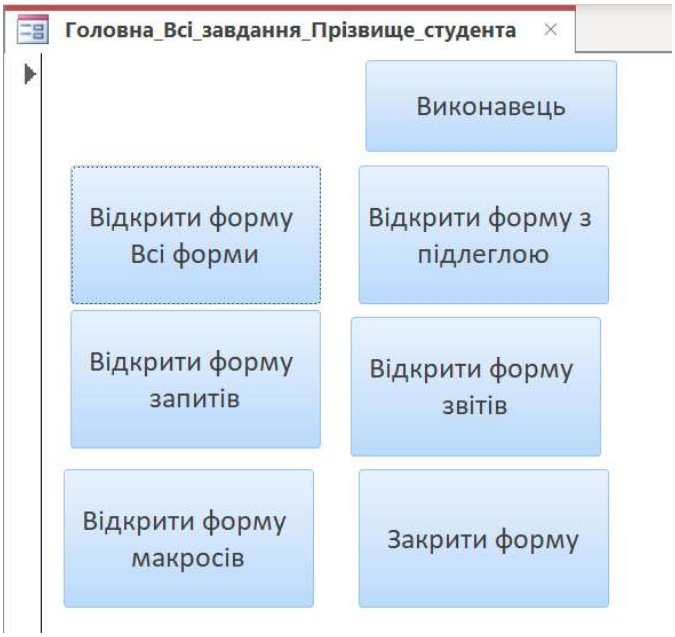



Рис. 7.23. Вигляд головної форми, яка має відкриватися при запуску застосування

2. Змінити стиль створених на комп'ютерному практикумі №6 форм для таблиць індивідуальних варіантів. Відключити стандартну панель для переходу між записами, а замість неї створити чотири кнопки (кнопки: попередній запис, наступний запис, на останній запис, на перший запис). Використати на формах об’єкти: поле зі списком, список, група перемикачів. Створити кнопку Додати та відредагувати для неї відповідну процедуру.

3. Для форми Склад_обладнання створити маску вводу для контролю введення даних у полі Дата_поставки.

4. Створити форму для таблиці Виробники_обладнання за допомогою Автоформа: в столбец.

5. Для форми Виробники_обладнання для введення номеру телефону і факсу використовувати шаблон.

6. Для форми Виробники_обладнання поле Виробник перетворити у поле зі списком.

7. На формах Склад_обладнання та Виробники_обладнання відключити стандартну панель для переходу між записами, а замість неї створити чотири кнопки (кнопки: попередній запис, наступний запис, на останній запис, на перший запис). Створити кнопку Додати та відредагувати для неї відповідну процедуру.

8. На формі Назва_обладнання поле Тип обладнання перетворити у поле із списком.

9. Дати письмові відповіді на контрольні запитання.

## 7.7. Контрольні запитання

1. Як прибрати стандартну панель переходу між записами?
2. Як перетворити поле у поле зі списком?
3. Які стандартні кнопки використовуються на формі для роботи з записами бази даних?
4. Чи може поле зі списком включати значення декількох полів запису?
5. Чи можна за допомогою одного поля зі списком відразу вибрати значення і ввести їх в кілька полів?
6. В якому випадку при створенні поля зі списком не вдалося підключитися майстер?
7. Чи можна для поля зі списком користуватися командами пошуку і сортування?
8. Де зберігається інформація про джерело даних для поля зі списком?
9. Де зберігається ім'я поля, в яке має бути введені значення, вибрані в списку?
10. Яка властивість призводить до відкриття форми в режимі додавання записів, при якому не можна переглядати раніше введених записів?

# Комп'ютерний практикум № 8. Використання макросів для розв’язку задач

*Мета роботи* – вивчити можливості створення та використання макросів для виконання дій за допомогою елементів керування, зв’язаних з відповідними макросами.

**Теоретичні відомості**

В комп'ютерному практикумі № 6 було розглянуто призначення макросів та конструктор для створення макросів. Надані приклади створення простих макросів для реалізації події **Нажатие кнопки** елементу керування **Кнопка**. Але в Access макрос може виконуватися у відповідь на численні події, якими наділені багато об'єктів Access. В СУБД Access можна визначити макрос, що виконує практично дії, які можна реалізувати натисканням клавіш на клавіатурі або за допомогою миші. Основною перевагою макросів є те, що вони можуть виконуватися у відповідь на багато видів подій. Прикладами подій є: зміна даних в полі, відкриття або закриття форми або звіту, натискання кнопки у формі і просто передача фокусу від одного поля до іншого. Завдяки зв'язку макросів з подіями можна автоматизувати роботу застосування, використовуючи макроси для відкриття форм, друку звітів, виконання послідовності запитів і дій, що залежать від значень деякого поля в базі даних, виведення повідомлень користувача або відключення попереджувальних повідомлень під час виконання запитів дії і багато чого іншого.

За допомогою макросів реалізуються алгоритми вирішення окремих задач для роботи з даними. Механізм зв'язування макросів з подіями в об'єктах дозволяє об'єднати процес вирішення розрізнених завдань в єдиний комплекс. Користувач, виконуючи різні дії в формах чи звітах, ініціює виконання макросів, що автоматизують розв’язок завдань, які безпосередньо пов'язані з цими діями.

## 8.1. Призначення макросів

Макроси можуть запускатися на виконання прямо з вікна бази даних. Існує також можливість використання ряду взаємозалежних макросів, головний

з яких користувач запускає з вікна бази даних, а далі все керування виконанням завдання здійснюється зсередини макросу. Макрос сам відкриває потрібні об'єкти, вибирає й обробляє дані, викликає інші макроси, дотримуючись алгоритму рішення завдання. При необхідності з макросу може бути ініційований діалог з користувачем. Для переходу по різних розгалуженнях макросу використовується умова, зазначена в рядку макрокоманди.

Макрос складається з послідовності макрокоманд. *Макрокоманда* – це інструкція, орієнтована на виконання певної простої дії.

Макрокомандою можна відкрити форму, звіт, надрукувати звіт, запустити на виконання запит, застосувати фільтр, привласнити значення, створити користувальницьке меню або панель команд. Макрокоманда **ВыполнитьКоманду** (RunCommand) дозволяє виконати будь-яку вбудовану команду Access, що виводиться в меню, на панелі інструментів або в контекстному меню. Наявний в Access набір макрокоманд реалізує практично будь-які дії, які необхідні при розробці невеликих персональних застосувань користувача.

У Microsoft Access існують різні типи макрокоманд, що дозволяють автоматизувати роботу застосування. Макроси можна використовувати для виконання наступних дій:

- для відкриття або закриття будь-якої таблиці, запиту, форми або звіту в будь-якому доступному режимі;
- для відкриття звіту в режимі попереднього перегляду або безпосереднього виведення звіту на принтер. Дані звіту можна вивести у файл у форматі RTF (.rtf), у звичайному текстовому форматі (.txt) або у файл Microsoft Excel (.xls) і потім відкрити їх у Microsoft Word, Windows Notepad або в Microsoft Excel;
- для виконання запиту на вибірку або запиту на зміну. При цьому параметри запиту можуть використовувати значення елементів керування будь-якої відкритої форми;
- для виконання дій у залежності від значень у базі даних, формі або звіті. Макроси можуть запускати інші макроси або процедури VBA. За допомогою макросу можна перервати виконання поточного макросу або

всіх макросів, скасувати подію, що запустила макрос, або навіть вийти з застосування;

- для встановлення значення будь-якого елемента керування форми або звіту. Можна емулювати роботу з клавіатурою і передавати дані, що вводяться з клавіатури, у системні вікна діалогу. За допомогою макросів можна обновляти значення в будь-яких елементах керування, джерелом даних яких є запит;
- для застосування фільтра, переходу до будь-якого запису і пошуку даних у базовій таблиці або запиті форми;
- для визначення спеціального рядка меню, що заміщає стандартне меню. Можна зробити доступним або не доступним, зняти галочку або позначити пункт спеціального меню, у тому числі і контекстного. Крім того, за допомогою макросів можна відкрити або закрити кожну зі стандартних або спеціальних панелей інструментів;
- для виконання будь-якої команди будь-якого меню Access;
- для переміщення, зміни розмірів, згортання або відновлення будь-якого вікна усередині робочої області Access ;
- для виведення на екран інформаційних подій і подачі звукових сигналів для залучення уваги до ваших повідомлень. Можна також відключати деякі попереджуючі повідомлення під час виконання запитів на зміни;
- для перейменування будь-якого об'єкта бази даних, копіювання обраного об'єкта в поточну або іншу базу даних Access. Можна використовувати макроси для збереження або видалення об'єктів з бази даних;
- для запуску застосування, а також для обміну даними з додатком за допомогою механізму DDL або буфера обміну. Існує можливість вивести дані з таблиці, форми, запиту або звіту у вихідний файл і відкрити його у відповідному застосуванні.

## 8.2. Кнопки головної форми

Для полегшення роботи з застосуванням на комп’ютерному практикумі №7 було створено головну форму (рис. 8.1), яка буде автоматично розкриватися

при кожному запуску бази даних та в якій будуть розташовані кнопки для виклику всіх об’єктів чи груп об’єктів, для реалізації можливостей створеного застосування.

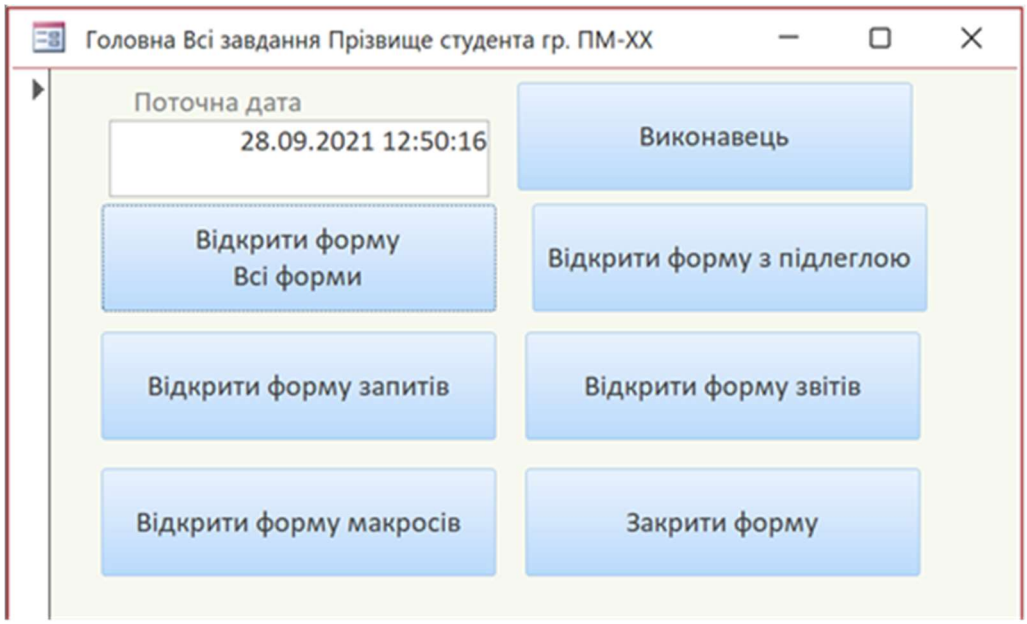



Рис. 8.1. Головна форма бази даних

При натисканні кожної з кнопок цієї форми, за виключенням кнопки **Закрити форму**, мають відкриватися об’єкти (форми різного вигляду, таблиці, форми, запити, макроси, звіти), які дозволяють виконувати дії з об’єктами даної групи. Наприклад, натискання кнопки **Відкрити форму макросі** розкриває форму з кнопками для виконання індивідуального завдання студента цього комп’ютерного практикуму (рис. 8.2). Кількість кнопок на цій формі, їх підписи та макроси, які реалізують відповідні завданню події, мають відповідати варіанту індивідуального завдання студента.

Подія натискання на будь-яку кнопку форми Індивідуальне завдання макроси варіант № Прізвище студента має виконувати попередньо створений макрос для реалізації відповідного завдання комп’ютерного практикуму.

Наприклад, кнопка **Послідовне виконання макросів** повинна за допомогою створеного макросу виконувати послідовно запити (взяти декілька запитів). Кнопка **Використання діалогово вікна для виконання дій** має організувати з використанням макросу, наприклад, відкриття деяких таблиць з використанням діалогового режиму роботи. Кнопка **Керування**

**відображенням** має керувати властивістю відображення (видиме/невидиме) елемента форми з використанням макросу.

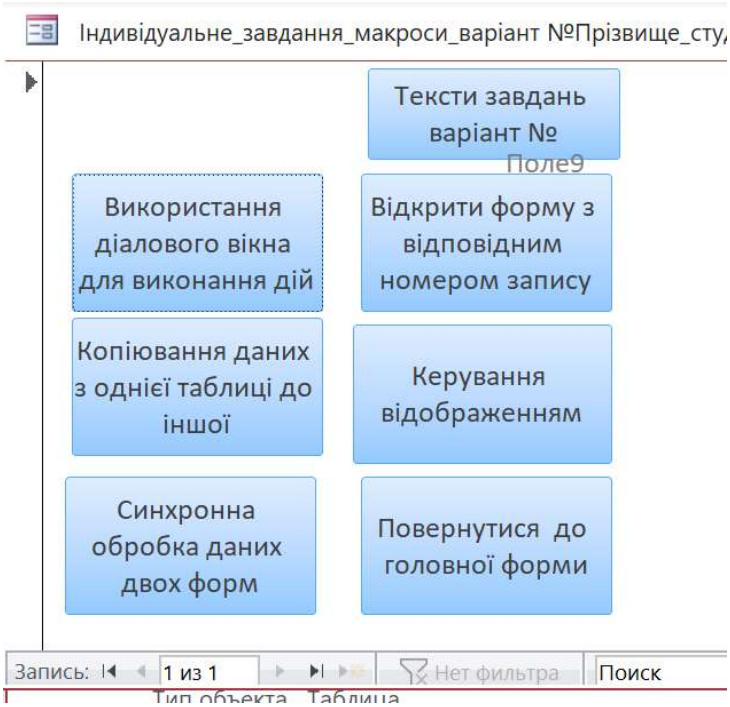



Рис. 8.2 Вигляд форми з кнопками, які реалізують завдання індивідуального варіанту комп'ютерного практикуму №8

Розглянемо основні прийоми створення макросів на прикладі завдання, що вимагає послідовного виконання дій при натисканні на відповідні кнопки та відображення результатів їх виконання. Макроси будуть запускатися при виборі кнопок форм.

## 8.3. Майстер створення виразів

Майстер створення виразів **Построитель выражений** призначений для спрощення запису виразів. Запустити **Построитель выражений** можна практично з будь-якого місця в додатку Microsoft Access, де виникає необхідність створення виразів. У таблицях і полях, в запитах, у властивостях форм і звітів, в елементах управління, запитах і макросах - у всіх цих елементах використовуються вирази для обчислень з даними і управління поведінкою програми. **Построитель выражений** надає простий доступ до імен полів і елементів управління в базі даних, а також до багатьох вбудованих функцій, доступних при написанні виразів. За допомогою **Построителя выражений**

можна як створювати вирази з нуля, так і вибирати готові вирази для виведення, наприклад, номерів сторінок, поточної дати або дати і часу [1].

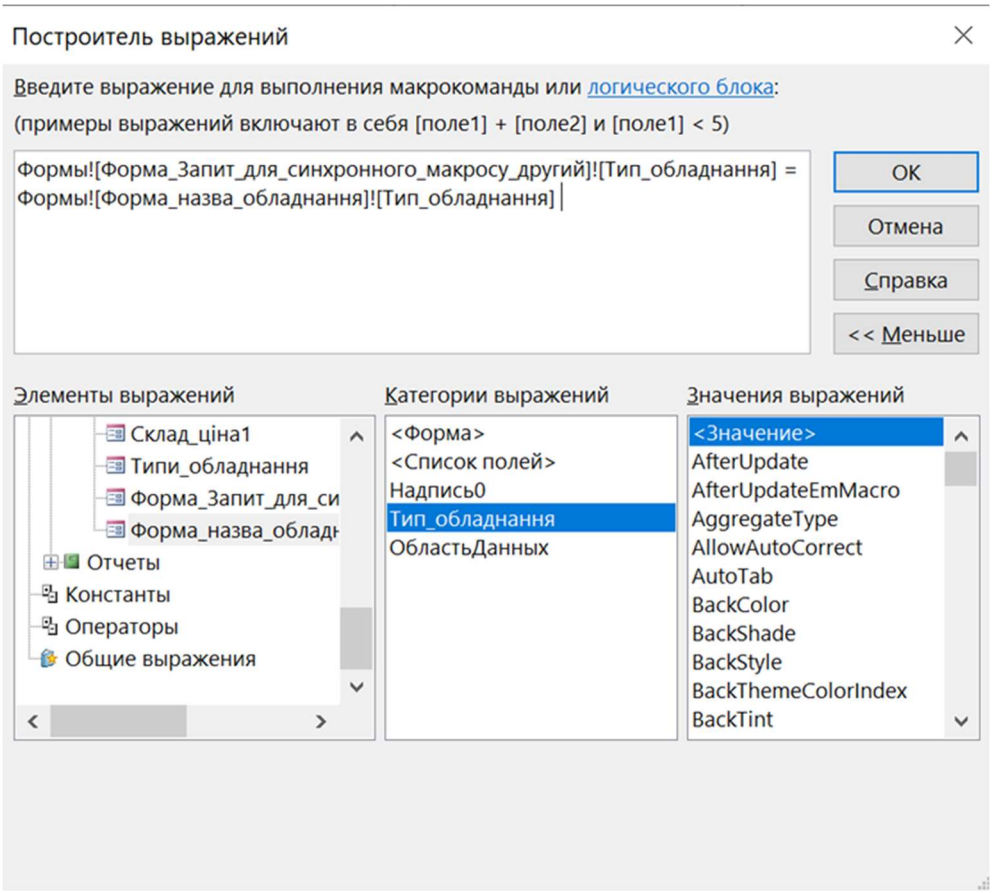



Рис. 8.3. Вигляд вікна Построитель выражений

## 8.4. Перевірка роботи макросу

Деякі макроси можуть бути запущені безпосередньо з вікна бази даних або вікна макросу, оскільки вони не залежать від елементів керування відкритої форми або звіту. Якщо макрос залежить від якої-небудь форми або звіту, його треба зв'язати з відповідною подією і запускати при виникненні цієї події. Перед запуском макросу непогано перевірити його роботу, виконавши макрокоманди в покроковому режимі.

Щоб почати покрокову перевірку, перейдіть у вікно бази даних, на вкладці **Макросы** виділіть ім'я макросу, який потрібно протестувати, і натисніть кнопку **Конструктор**. Після відкриття вікна макросу натисніть кнопку По кроках на панелі інструментів або виберіть команду **Запуск >По шагам.**

Тепер після запуску макросу Access буде відкривати вікно діалогу Покрокове виконання макросу перед виконанням кожного кроку. У цьому вікні ви побачите ім'я макросу, назву макрокоманди, умови її виконання та аргументи макрокоманди.

Якщо під час виконання застосування в якому-небудь макросі виникла помилка, то Access спочатку виведе вікно діалогу, що пояснює її. Потім ви побачите схоже на Покрокове виконання макросу вікно діалогу: Помилка макрокоманди з інформацією про макрокоманду, що викликала помилку. У цей момент можна натиснути тільки кнопку **Прервать** і відредагувати макрос, щоб усунути причину помилки.

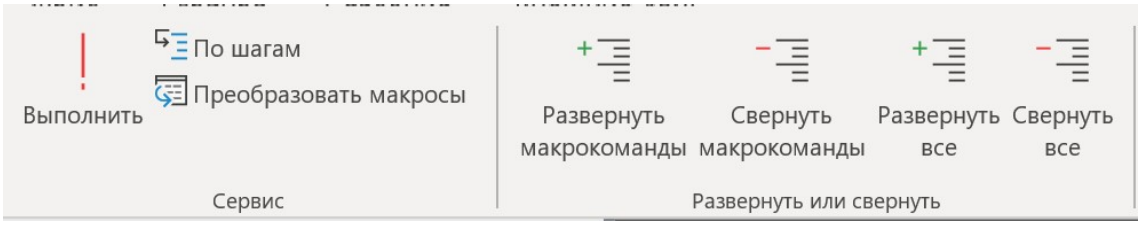



Рис. 8.4. Вигляд операцій для роботи з тестуванням макросів

## 8.5. Створення макросу для виконання дій у відповідності до умови

Створимо форму за зразком (рис. 8.5). В області даних розмістимо поле (свободное). Потім перетворимо його в поле із списком. Значення поля обмежуються списком з трьох елементів. Збережемо форму з ім’ям Форма_Вибір_таблиці_макросом_з_умовою.


Рис. 8.5 Вигляд форми Форма_Вибір_таблиці_макросом_з_умовою в режимі конструктора

Створимо макрос, який в залежності від вибору значення із списку, відкриватиме відповідну таблицю.

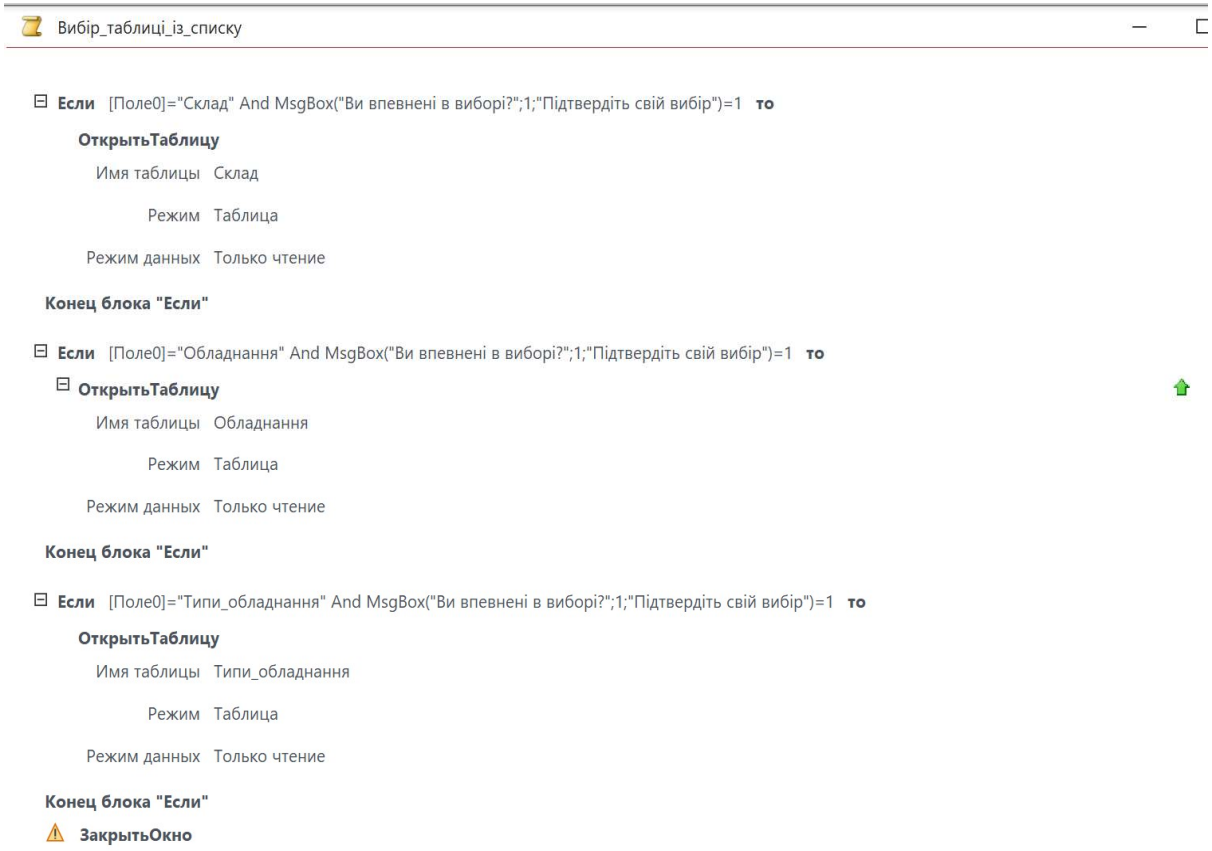



Рис. 8.6. Макрос для вибору даних таблиці обраної в полі зі списком

Макрос Вибір_таблиці_із_списку необхідно прив’язати до властивості поля із списком **После обновления**

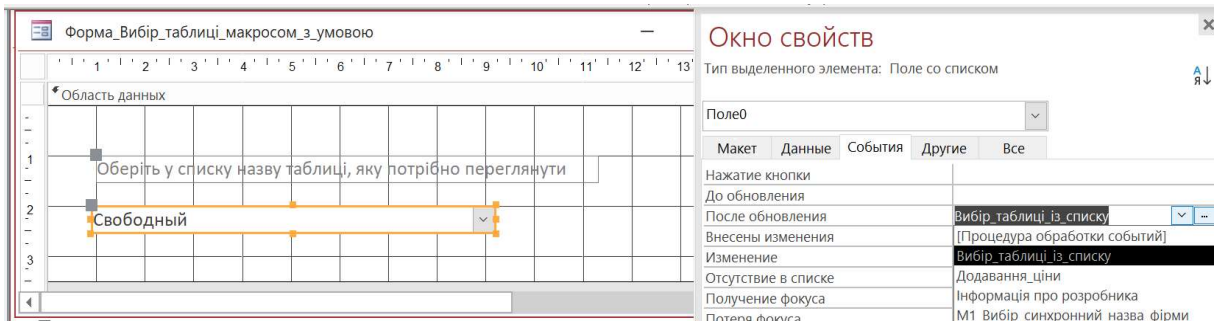



Рис. 8.7. Приєднання макроса до властивості поля із списком

При виборі імені таблиці із списку форми , виводиться інформаційне вікно для підтвердження вибраної дії та відкривається таблиця з відповідним ім’ям.

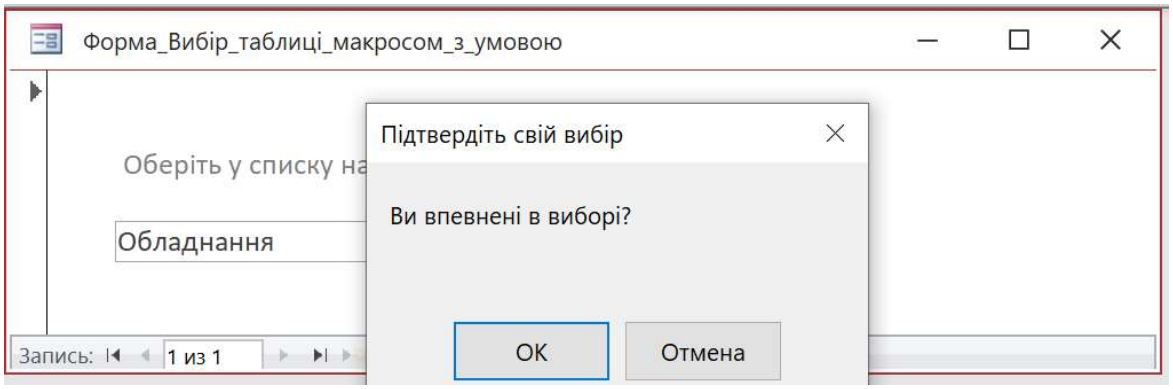



Рис. 8.8. Виведення інформаційного вікна для підтвердження вибраної дії

## 8.6. Створення макросу для синхронної обробки даних двох форм

Розглянемо створення макросів, призначених для відбору даних в одній формі, які будуть запускатися при ініціалізації події в іншій формі. Така організація роботи з формами називається синхронною роботою форм.

В яких випадках виникає потреба в синхронній обробці даних? Наприклад, необхідно отримати інформацію про виробників обраного типу обладнання.

Для реалізації такого завдання необхідно створити дві нові форми:

- Форма_назва_обладнання, на якій буде розташоване одне поле Тип_обладнання;
- Форма_Запит_для_синхронного макросу_другий, джерелом даних якої буде попередньо створений запит на вибірку (рис. 8.9).

Створимо форму Форма_Запит_для_синхронного_макросу_другий, джерелом даних якої має бути запит Запит_для_синхронного макросу_другий (рис. 8.10).

Після створення форм, приступимо до створення макросів. Для вирішення завдання створимо макроси з іменами Макрос_синхронна_обробка_М1 (рис. 8.11) та Макрос_М2.

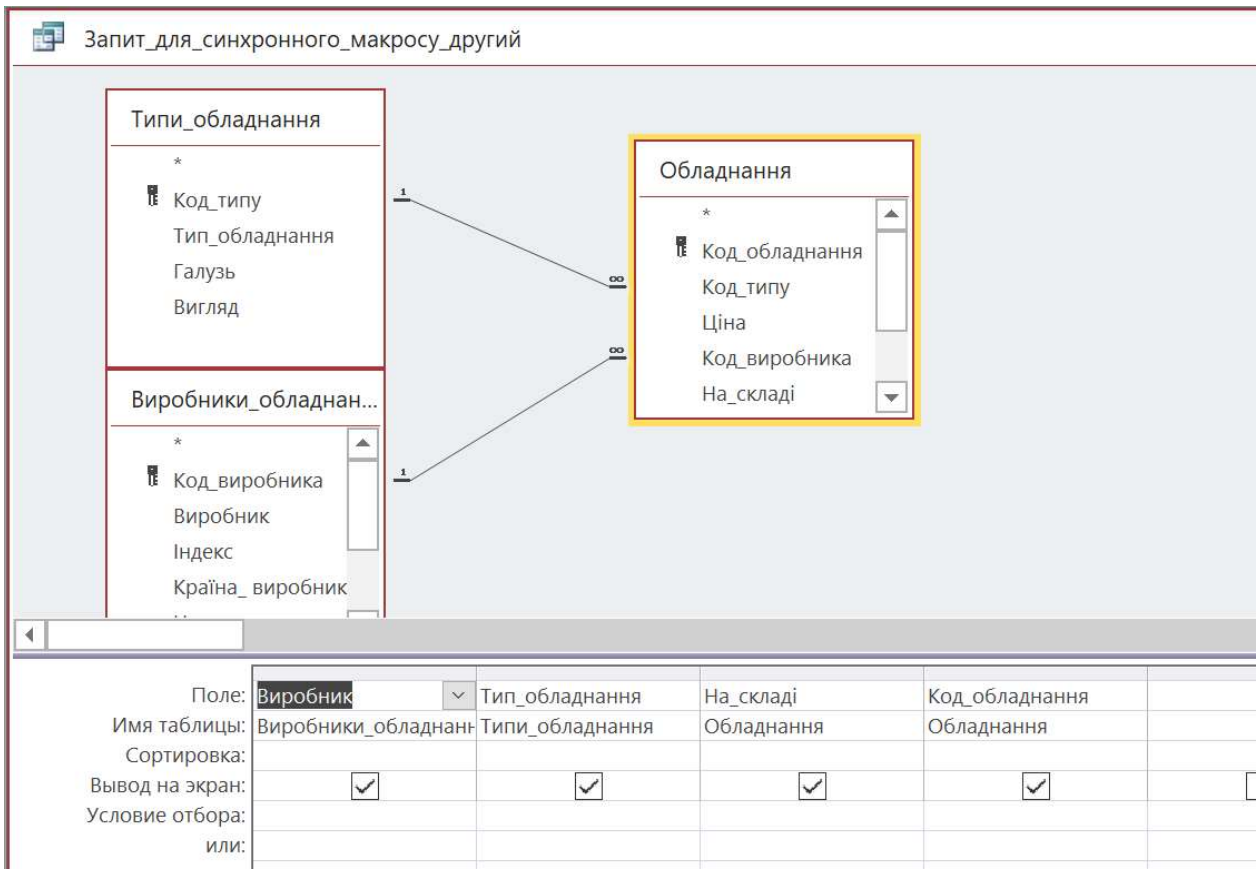



Рис. 8.9. Вигляд запиту Запит_для_синхронного_макросу_другий в режимі конструктора

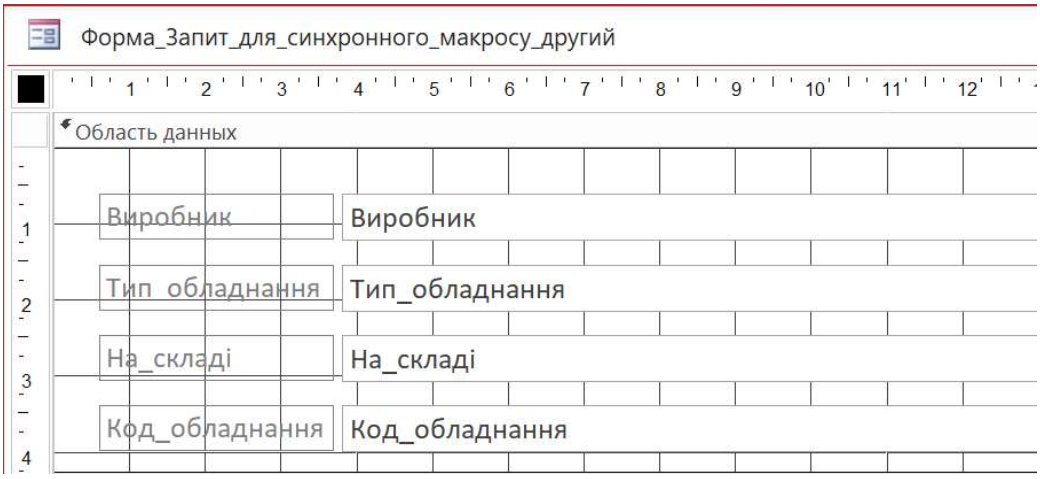



Рис. 8.10. Вигляд форми Форма_Запит_для_синхронного макросу_другий в режимі конструктора

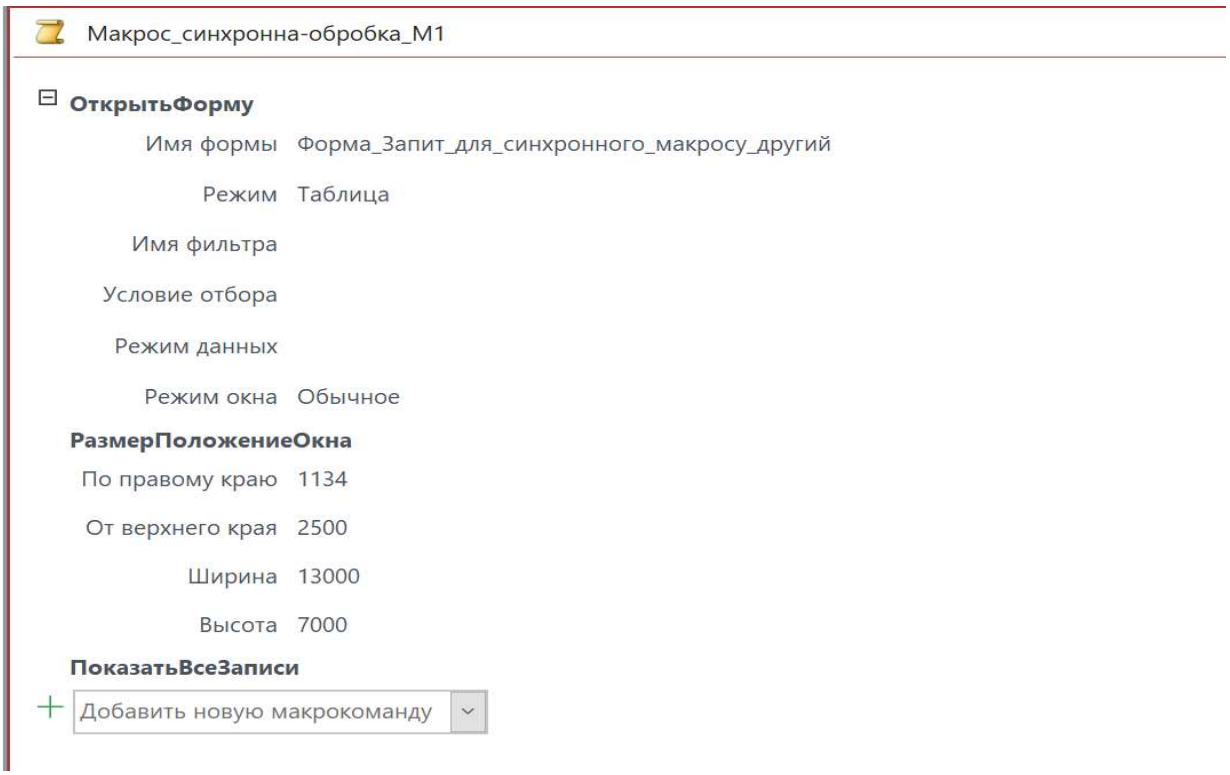



Рис. 8.11. Макрос_синхронна_обробка_М1 для відкривання форми Форма_Запит_для_синхронного макросу_другий списком

Макрос Макрос_синхронна_обробка_М1, який виконується при відкритті форми Форма_назва_обладнання, відкриває форму Форма_Запит_для_синхронного макросу_другий у режимі Таблиця (рис. 8.12). Пересування форми відносно лівого верхнього кута екрану визначається параметрами команди **РазмерПоложениеОкна** даного макросу.

Макрос Макрос_М2 фільтрує записи по назві обладнання за значенням поля Тип_обладнання, взятому з поточного запису форми Форма_назва_обладнання. Макрос повинен виконуватися тоді, коли користувач, працюючи в формі Форма_назва_обладнання, ініціює подію Вход для поля з назвою обладнання.

Для запуску макроса Макрос_синхронна_обробка_М1 необхідно встановити зв’язок макроса та події Открытие у властивостях форми Форма_назва_обладнання.

Для запуску макроса Макрос_М2 необхідно встановити зв’язок макроса та події Вход у властивостях поля Тип_обладнання форми Форма_назва_обладнання.

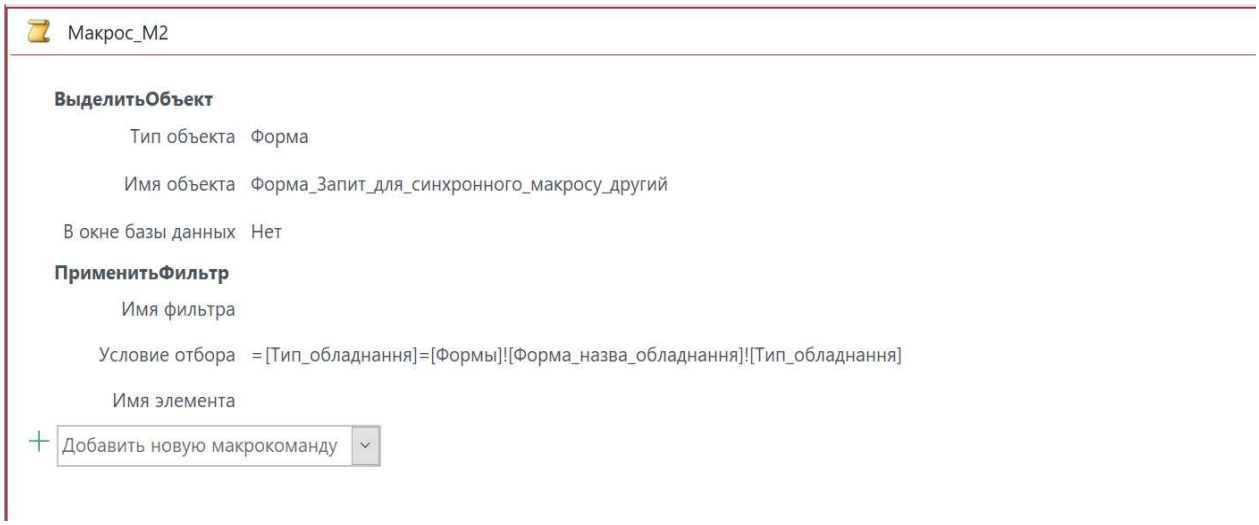



Рис. 8.12. Макрос_М2 для відкривання форми Форма_Запит_для_синхронного макросу_другий з даними, що відповідають критерію відбору

Форма Форма_назва_обладнання запускається на виконання кнопкою «Синхронна обробка даних двох форм» форми Всі_макроси (рис. 8.13). Макрос що запускається на виконання подією **Нажатие кнопки** має вигляд

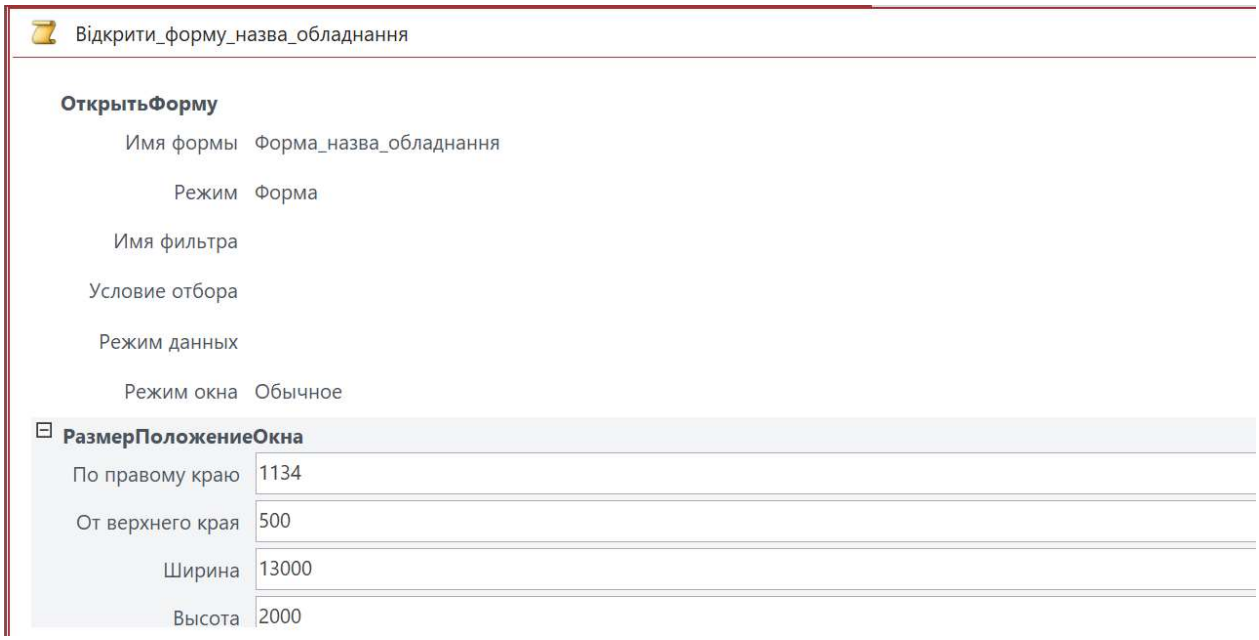



Рис. 8.13. Макрос_М2 для відкривання форми Форма_назва_обладнання

Тепер після запуску форми Форма_назва_обладнання буде відкриватися форма Форма_Запит_для_синхронного макросу_другий з назвами фірм виробників обладнання вибраного у полі Тип_обладнання форми Форма_назва_обладнання. Приклад виконання представлено на рис. 8.14.

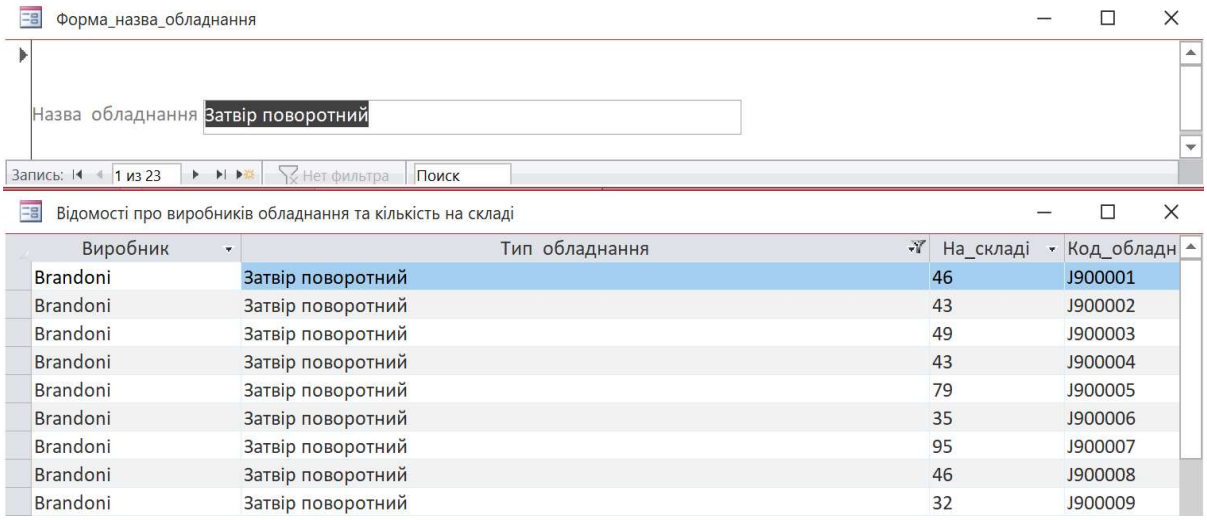



Рис. 8.14. Результат виконання події Вход текстового поля Тип_обладнання форми Форма_назва_обладнання

## 8.7. Копіювання даних з однієї таблиці до іншої з використанням макроса

Розглянемо копіювання даних з однієї таблиці до іншої з використанням макроса. В цьому прикладі за один раз буде копіюватися весь запис. Щоб реалізувати таку задачу необхідно мати дві таблиці з однаковою структурою. Одна таблиця, яка буде джерелом даних, заповнена даними. Друга буде порожньою і до неї будемо копіювати деякі обрані записи. У такий спосіб може бути реалізовано задачу підбору деяких товарів. Для прикладу скопіюємо таблицю Склад_ціна та збережемо її з ім’ям Склад_ціна_вибір. Скопіюємо лише структуру таблиці Склад_ціна та збережемо її з ім’ям Склад_ціна_копіювання. Для таблиці створимо форму Склад_ціна_вибір та розташуємо на ній кнопку. Підпишемо кнопку «Копіювати» (рис. 8.15).

Створимо макрос для копіювання даних з таблиці Склад_ціна_вибір до таблиці Склад_ціна_копіювання (рис. 8.16).

***Зауваження!*** Якщо виникають проблеми з деякими командами, наприклад, Копировать, то необхідно активізувати закладку Показать все действия в режимі роботи з макросами. Після чого відкриється доповнений список макрокоманд.

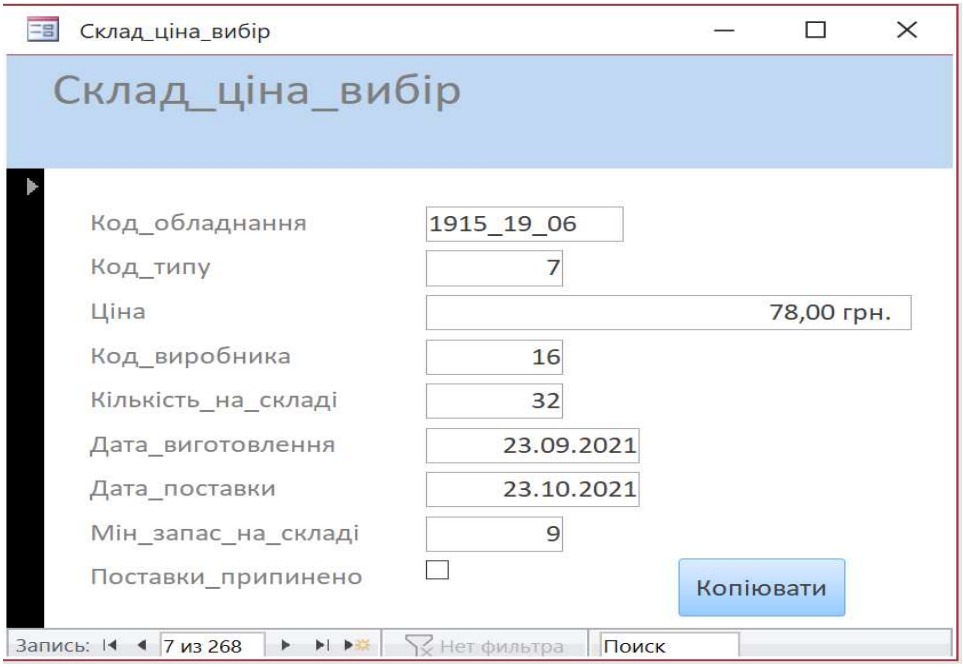



Рис. 8.15. Вигляд форми Склад_ціна_вибір

```
/*   Виділити запис у формі Склад_ціна_вибір
   ЗапускКомандыМеню
         Команда   ВыделитьЗапись
/*   Копіювати
   ЗапускКомандыМеню
         Команда   Copy
/*   Відкрити таблицю Склад_ціна_копіювання
   ОткрытьТаблицу
      Имя таблицы   Склад_ціна_копіювання
            Режим   Таблица
     Режим данных   Изменение
/*   Проініціалізувати   таблицю Склад_ціна_копіювання для додавання нового запису
   НаЗапись
       Тип объекта   Таблица
      Имя объекта   Склад_ціна_копіювання
            Запись   Новая
         Смещение
/*   Виділити запис
   ЗапускКомандыМеню
         Команда   ВыделитьЗапись
/*   Вставити запис
   ЗапускКомандыМеню
         Команда   Paste
```

Рис. 8.16. Вигляд макроса МАКРОС_ВИБІР_Склад_ціна

Тепер до події **Нажатие кнопки** кнопки **Копіювати** потрібно приєднати макрос МАКРОС_ВИБІР_Склад_ціна. Після натисканння кнопки **Копіювати** до таблиці Склад_ціна_копіювання буде додано один запис.

## 8.8. Завдання до комп'ютерного практикуму

1. Модифікувати головну форму, створену на комп'ютерних практикумах 6-7 у відповідності до форми (рис. 8.1). Кнопка **Відкрити форму макросів** відкриває форму другого рівня (рис. 8.2), на якій розташовані кнопки для виконання макросів створених у відповідності до індивідуального завдання (табл. 8.1).
2. Для успішного виконання індивідуального завдання необхідно уважно прочитати теоретичі відомості, в яких розглянуто використання макросів для більшості завдань, представлених в табл.8.1

Таблиця 8.1

| Варіант | Перелік макросів, які мають бути створені та запускатися на виконання подіями |
|---|---|
| 1 | 1) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Виробники обладнання. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей)<br>2)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Виробник – джерелом даних буде таблиця Виробники обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання) та Виробник. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>3)Макрос пересунути таблицю при відкриванні відносно початкових координат екрану(супроводжується звуковим сигналом).<br>4)Макрос для відкриття форми Лічильники з заданим записом (запис 7).<br>5)Керування відображенням елемента керування у формі Лічильники з використанням макросу. |

| | |
|---|---|
| 2 | 1)Макрос, який в залежності від вибору значення із списку з трьох значень, відкриватиме відповідну форму .<br>2)Макрос, що пересуває кожну форму при відкриванні відносно початкових координат екрану(супроводжується звуковим сигналом).<br>3) Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Тип_обладнання – джерелом даних буде таблиця Типи_обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання) та Країна_виробника. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>4) Керування відображенням елемента керування у формі Затвори_клапани з використанням макросу.<br>5) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Типи_обладнання. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей) |
| 3 | 1)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Країна_виробника – джерелом даних буде таблиця Виробники_обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання), Виробник. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>2) Макрос для переходу до 5-го запису у формі Типи_обладнання<br>3) Керування відображенням елемента керування у формі Затвори_клапани з використанням макросу.<br>4) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Виробники_обладнання. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей)<br>5) Макрос, який в залежності від вибору значення із списку з трьох значень, запускатиме на виконання відповідний запит |

| | |
|---|---|
| 4 | 1 )Макрос, який в залежності від вибору значення із списку з трьох значень, відкриватиме відповідну таблицю: Типи_обладнання, Виробники_обладнання, Лічильники .<br>2) Макрос пересунути таблицю при відкриванні відносно початкових координат екрану(супроводжується звуковим сигналом).<br>3)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Виробник – джерелом даних буде таблиця Виробники_обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання) та Ціна. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>4) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Обладнання. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей)<br>5) Керування відображенням елемента керування у формі Виробники_обладнання з використанням макросу. |
| 5 | 1)Макрос для відкриття форми Крани_фільтри з заданим записом (запис 8).<br>2)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Виробник – джерелом даних буде таблиця Виробники обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання) та Виробник. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>3)Керування відображенням елемента керування у формі Лічильники з використанням макросу.<br>4) Макрос, який в залежності від вибору значення із списку з трьох значень, відкриватиме відповідну таблицю: Обладнання, Виробники_обладнання, Крани_фільтри .<br>5) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Крани_фільтри. (В якості зразка використати приклад розділу 8.7Теоретичних відомостей) |

| | |
|---|---|
| 6 | 1)Макрос, який в залежності від вибору значення із списку з трьох значень, відкриватиме відповідну форму .<br>2)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Тип_обладнання – джерелом даних буде таблиця Типи_обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання) та Країна_виробника. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>3) Макрос для переходу до 5-го запису у формі Типи_обладнання<br>4) Керування відображенням елемента керування у формі Затвори_клапани з використанням макросу.<br>5) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Типи_обладнання. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей) |
| 7 | 1) Макрос, який в залежності від вибору значення із списку з трьох значень, відкриватиме відповідну таблицю .<br>2)Керування відображенням елемента керування у формі Лічильники з використанням макросу.<br>3)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Країна_виробника – джерелом даних буде таблиця Виробники_обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання), Виробник. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>4) Макрос для переходу до 5-го запису у формі Типи_обладнання<br>5) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Виробники_обладнання. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей) |

| | |
|---|---|
| 8 | 1)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Виробник – джерелом даних буде таблиця Виробники_обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання) та Ціна. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>2 )Макрос, який в залежності від вибору значення із списку з трьох значень, відкриватиме відповідну таблицю: Типи_обладнання, Виробники_обладнання, Лічильники .<br>3) Макрос, що пересуває кожну форму при відкриванні відносно початкових координат екрану(супроводжується звуковим сигналом).<br>4) Керування відображенням елемента керування у формі Виробники_обладнання з використанням макросу<br>5) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Затвори_клапани. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей) |
| 9 | 1)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Виробник – джерелом даних буде таблиця Виробники обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання) та Виробник. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>2)Керування відображенням елемента керування у формі Лічильники з використанням макросу.<br>3) Макрос, який в залежності від вибору значення із списку з трьох значень, відкриватиме відповідну таблицю: Типи_обладнання, Лічильники, Обладнання<br>4)Керування відображенням елемента керування у формі Обладнання з використанням макросу.<br>5) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Обладнання. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей) |

| | |
|---|---|
| 10 | 1)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Тип_обладнання – джерелом даних буде таблиця Типи_обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання) та Країна_виробника. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>2)Макрос, який в залежності від вибору значення із списку з трьох значень, відкриватиме відповідну таблицю .<br>3)Макрос для переходу до 4-го запису у формі Виробники_обладнання<br>4) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Крани_фільтри. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей)<br>5)Керування відображенням елемента керування у формі Крани_фільтри з використанням макросу. |
| 11 | 1)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Країна_виробника – джерелом даних буде таблиця Виробники_обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання), Виробник. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>2 )Макрос, який в залежності від вибору значення із списку з трьох значень, відкриватиме відповідну таблицю: Типи_обладнання, Виробники_обладнання, Лічильники .<br>3)Керування відображенням елемента керування у формі Крани_фільтри з використанням макросу.<br>4)Макрос для послідовного виконання запитів (взяти декілька запитів) і виконання організувати з використанням діалогового режиму роботи<br>5) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Крани_фільтри. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей) |

| | |
|---|---|
| 12 | 1)Макрос, який в залежності від вибору значення із списку з трьох значень, відкриватиме відповідну таблицю: Типи_обладнання, Виробники_обладнання, Крани_фільтри .<br>2)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Виробник – джерелом даних буде таблиця Виробники_обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання) та Ціна. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>3)Макрос послідовного виконання запитів<br>4)Керування відображенням елемента керування у формі Крани_фільтри з використанням макросу<br>5) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Затвори_клапани. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей) |
| 13 | 1)Керування відображенням елемента керування у формі Лічильники з використанням макросу.<br>2)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Виробник – джерелом даних буде таблиця Виробники обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання) та Виробник. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>3) Макрос для послідовного виконання запитів (взяти декілька запитів) і виконання організувати з використанням діалогового режиму роботи<br>4) Макрос для переходу до 3-го запису у формі Виробники_обладнання<br>5) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Лічильники. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей). |

| | |
|---|---|
| 14 | 1)Макрос, який в залежності від вибору значення із списку з трьох значень, відкриватиме відповідну форму .<br>2)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Тип_обладнання – джерелом даних буде таблиця Типи_обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання) та Країна_виробника. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>3) Макрос для переходу до 4-го запису у формі Затвори_клапани<br>4)Керування відображенням елемента керування у формі Крани_фільтри з використанням макросу<br>5) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Затвори_клапани. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей). |
| 15 | 1)Макрос для послідовного виконання запитів (взяти декілька запитів) і виконання організувати з використанням діалогового режиму роботи<br>2)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Країна_виробника – джерелом даних буде таблиця Виробники_обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання), Виробник. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>3) Керування відображенням елемента керування у формі Крани_фільтри з використанням макросу.<br>4) Макрос, який в залежності від вибору значення із списку з трьох значень, відкриватиме відповідну форму .<br>5) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Крани_фільтри. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей) |

| | |
|---|---|
| 16 | 1)Макрос синхронної обробки даних двох форм. Створити дві нові форми для синхронної роботи з даними цих форм. В першій формі має бути одне поле Виробник – джерелом даних буде таблиця Виробники_обладнання. Джерелом даних другої форми буде запит з полями Тип_обладнання (назва обладнання) та Ціна. Для створення запиту використати таблиці, Типи_обладнання, Обладнання та Виробники_обладнання.<br>2 )Макрос, який в залежності від вибору значення із списку з трьох значень, відкриватиме відповідну таблицю: Типи_обладнання, Виробники_обладнання, Лічильники .<br>3)Макрос послідовного виконання запитів<br>4)Керування відображенням елемента керування у формі Крани_фільтри з використанням макросу<br>5) Виконати копіювання даних з однієї таблиці до іншої з використанням макроса для таблиці Затвори_клапани. (В якості зразка використати приклад розділу 8.7 Теоретичних відомостей) |

## 8.9. Контрольні запитання

1. Яка команда використовується для керування відображенням елемента керування?
2. Сформулюйте порядок дій при створенні макросу за допомогою миші?
3. Як автоматизувати послідовне виконання запитів дії і як включити текст попереджуючих повідомлень?
4. Як управляти розміщенням об'єктів на екрані?
5. Які способи запуску макросу на виконання пропонує Access?
6. Як зв'язати виконання макросу з подією?
7. Як використовувати дані поточних записів різних об'єктів?
8. Як поєднати в групу макроси, пов'язані з розв'язанням одного завдання?

Додаток

| Категория | Назначение | Макрокоманда |
|---|---|---|
| Работа *с* | Отбор данных | Применить Фильтр (ApplyFilter) |
| | Перемещение по данным | Следующая Запись (FindNext) Найти Запись (FindRecord)<br>К Элементу Управления (GoToControl)<br>На Страницу (GoToPage)<br>На Запись (GoToRecord) |
| | Обновление данных или экрана | Обновление (Requery) Показать Все Записи (ShowAIIRecords) |
| Выполнение | Выполнение команды | Выполнить Команду (RunCommand) |
| | Выполнение макроса, процедуры или запроса | Запуск Макроса (RunMacro)<br>Запуск Программы (RunCode) Открыть Запрос (OpenQuery) Запуск Запроса SQL (RunSQL) |
| | Выполнение другого приложения | Запуск Приложения (RunApp) |
| | Прерывание выполнения | Отменить Событие (CancelEvent)<br>Остановить Все Макросы (StopAIIMacros)<br>Остановить Макрос (StopMacro) |
| | Выход из Microsoft Access | Выход (Quit) |
| Импорт/экспорт | Передача объектов Microsoft Access в другие приложения | Вывести в Формате (OutputTo)<br>Отправить Объект (SendObject) |
| | Преобразование формата данных | Преобразовать Базу Данных (TransferDatabase)<br>Перенос Базы Данных SQL (TransferSQLDatabase)<br>Преобразовать Электронную<br>Таблицу (TransferSpreadsheet) Преобразовать Текст (TransferText) |
| Работа с объектами | Копирование, переименование и сохранение объекта | Копировать Объект (CopyObject)<br>Копировать Файл Баз Данных (CopyDatabaseFile)<br>Переименовать (Rename) Сохранить (Save) |
| | Удаление объекта | Удалить Объект (DeleteObject) |
| | Изменение размеров или положения окна | Развернуть (Maximize)<br>Свернуть (Minimize)<br>Сдвиг Размер (MoveSize) Восстановить (Restore) |

| | | |
|---|---|---|
| | Открытие и закрытие объекта | Открыть Форму (OpenForm) Открыть Функцию (OpenFunction) (OpenModule)<br>Открыть Запрос (OpenQuery) Открыть Отчет (OpenReport) Открыть Таблицу (ОрепТаЫе)<br>Открыть Страницу Доступа К Данным (OpenDataAccessPage) Открыть Схему (OpenDiagram)<br>Открыть Сохраненную Процедуру (OpenStoreProcedure) Открыть редставление (OpenView) Закрыть (Close) |
| | Печать объекта | Печать (Printout) |
| | Выделение объекта | Вылить Объект (SelectObject) |
| | Задание значения поля, элемента управления | Задать Значение (SetValue) |
| | Обновление объекта | Обновить Объект (RepaintObject) |
| Прочие | Создание специальной или общей строки меню, специального или глобального контекстного меню | Добавить Меню (AddMenu) |
| | Задание состояния пунктов меню в специальной или общей строке меню | Задать Команду Меню (SetMenultem) |
| | Вывод информации на экран | Вывод На Экран (Echo)<br>Песочные Часы (Hourglass) Сообщение (MsgBox)<br>Установить Сообщения (SetWarnings) |
| | Генерация нажатий клавиш | Команды Клавиатуры (SendKeys) |
| | Вывод на экран или скрытие встроенной или специальной панели инструментов | Панель Инструментов (ShowToolbar) |
| | Подача звукового сигнала | Сигнал (Beep) |

# Комп'ютерний практикум № 9. Створення форм з закладками

*Мета роботи* - набути навичок у створенні закладок на формі за допомогою конструктора СУБД Access.

## Теоретичні відомості

Створення закладок на формі допомагає зробити її більш впорядкованою та простішою в користуванні, особливо якщо форма містить багато елементів керування або на форму має буде виведено велику кількість даних. При розташуванні споріднених елементів керування на одній закладці форми, можна зберегти порядок у формі та зробити її зручнішою для роботи з даними.

Для додавання закладок на форму використовується елемент керування «Вкладка». Кожна сторінка цього елемента керування відіграє роль контейнера для інших елементів керування — написів, списків або кнопок. Процедуру додавання елемента керування «Вкладка» на форму описано у наступному прикладі..

**Приклад 9.1.**

1. За допомогою конструктора створити нову форму. Вибрати у якості джерела даних таблицю **КопіяЛічильники**. Для цього потрібно відкрити вікно **Окно Свойств і на** закладенці **Данные** в полі Источник Записей вибрити з випадаючого списку таблицю **КопіяЛічильники** (рис. 9.1).
2. Створити закладки натиснувши кнопку  на панелі інструментів.
3. Перейменувати першу закладку, записавши в полі **Подпись - Свойства Вкладки** ім'я закладки **Вибір1**, та розташувати на ній наступні поля: Тип, Умовний_прохід, Умовний_прохід, Перепад_тиску, Ціна. Перейменуйте поля відповідно: Тип, Умовний прохід від, Умовний прохід до, Перепад тиску до, Ціна до (рис. 9.2).

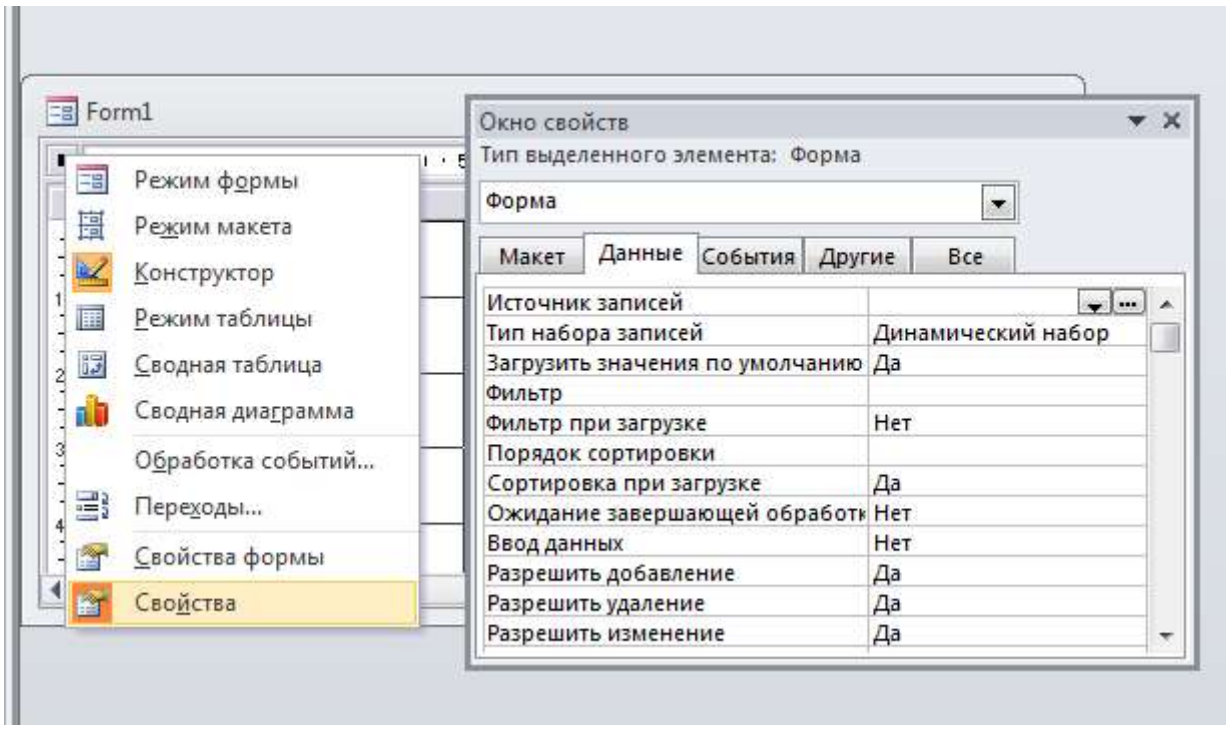



Рис. 9.1. Вибір джерела даних.( Таблиця **КопіяЛічильники**)

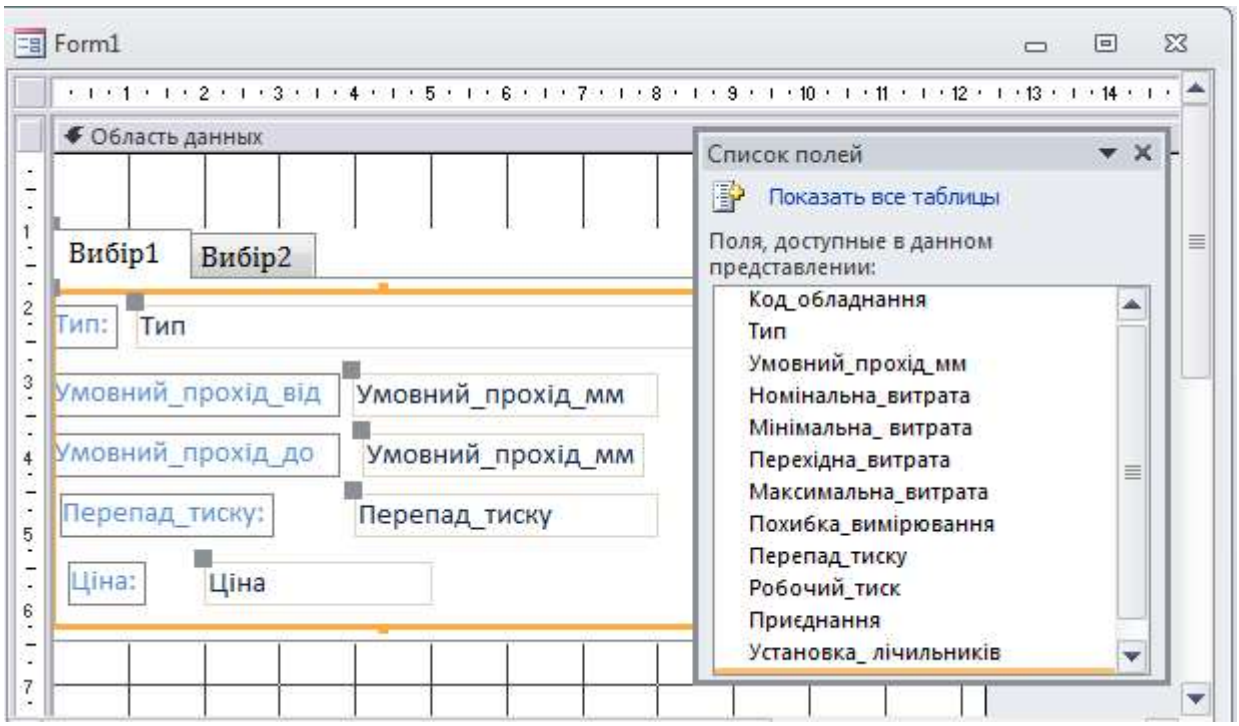



Рис. 9.2. Створення закладки Вибір1 у формі

4. Перейменувати другу закладку записавши в поле **Подпись - Свойства Вкладки** ім'я закладки **Вибір2**, та розташувати на ній наступні поля: Тип, Номінальна_витрата, Номінальна_витрата, Приєднання, Ціна. Перейменуйте поля відповідно: Тип, Номінальна витрата від, Номінальна витрата до, Приєднання, Ціна до (рис. 9.3).

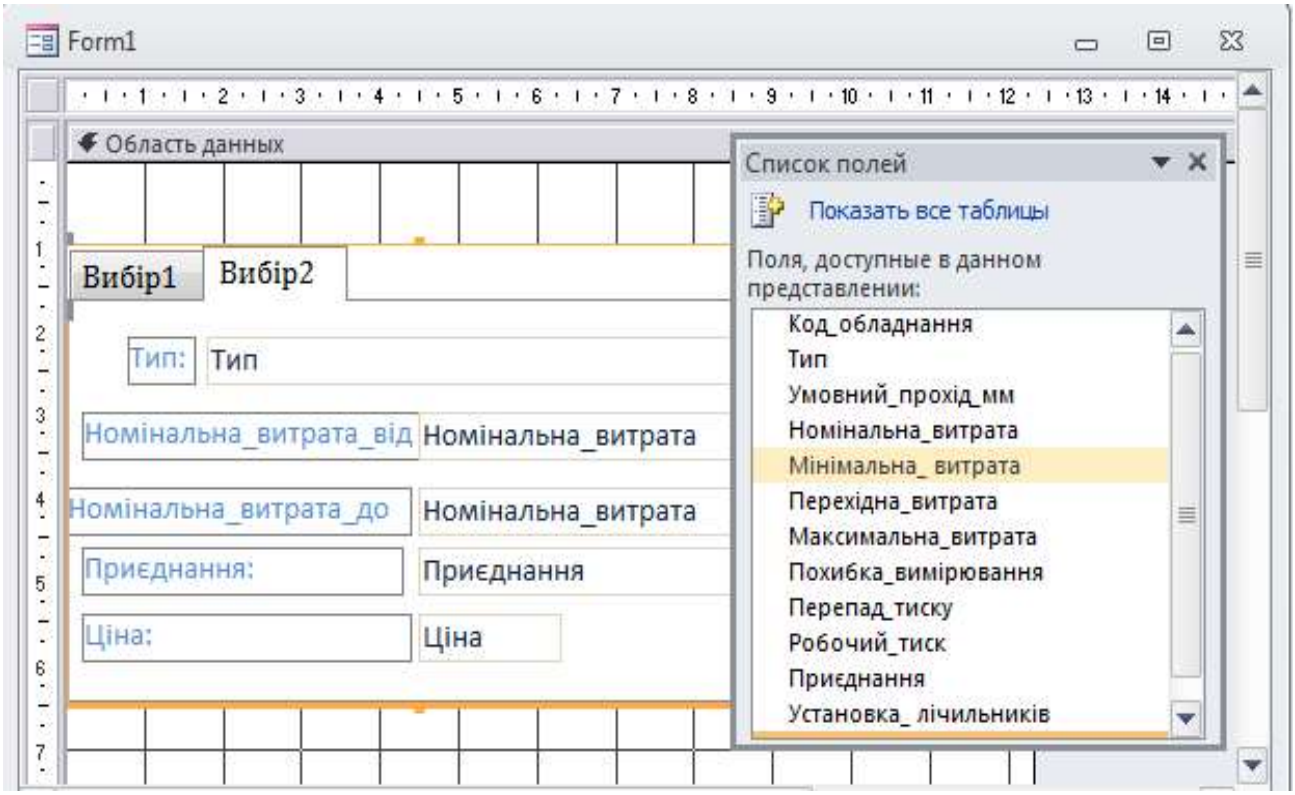



Рис. 9.3. Створення закладки Вибір2 у формі

5. Створити нову вкладку, натиснувши праву кнопку миші на закладці та вибрати **Вставить вкладку** (рис. 9.4).

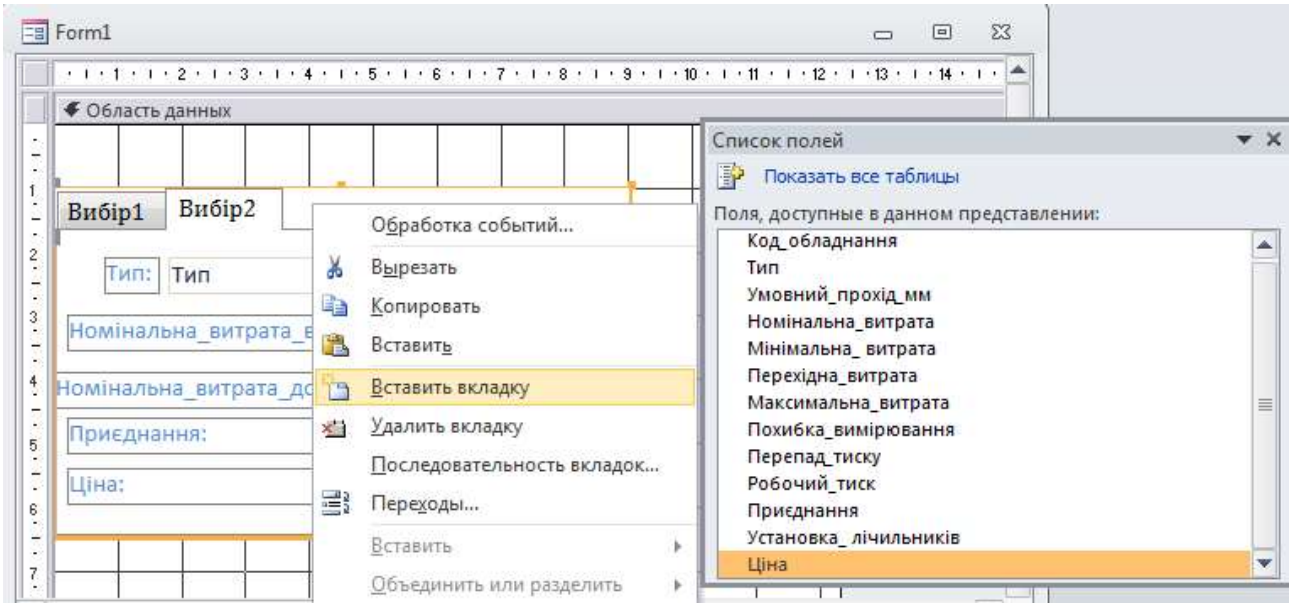



Рис. 9.4. Створення закладки Вибір3 у формі

6. Перейменувати третю закладку, записавши в полі **Подпись - Свойства Вкладки** ім'я закладки **Вибір3**, та розташувати на ній наступні поля: Тип, Установка_лічильників, Похибка_вимірювання, Робочий_тиск, Ціна.

Перейменуйте поля відповідно: Тип, Установка ліч., Похибка вимірювання, Робочий тиск, Ціна до.( рис 9.5).

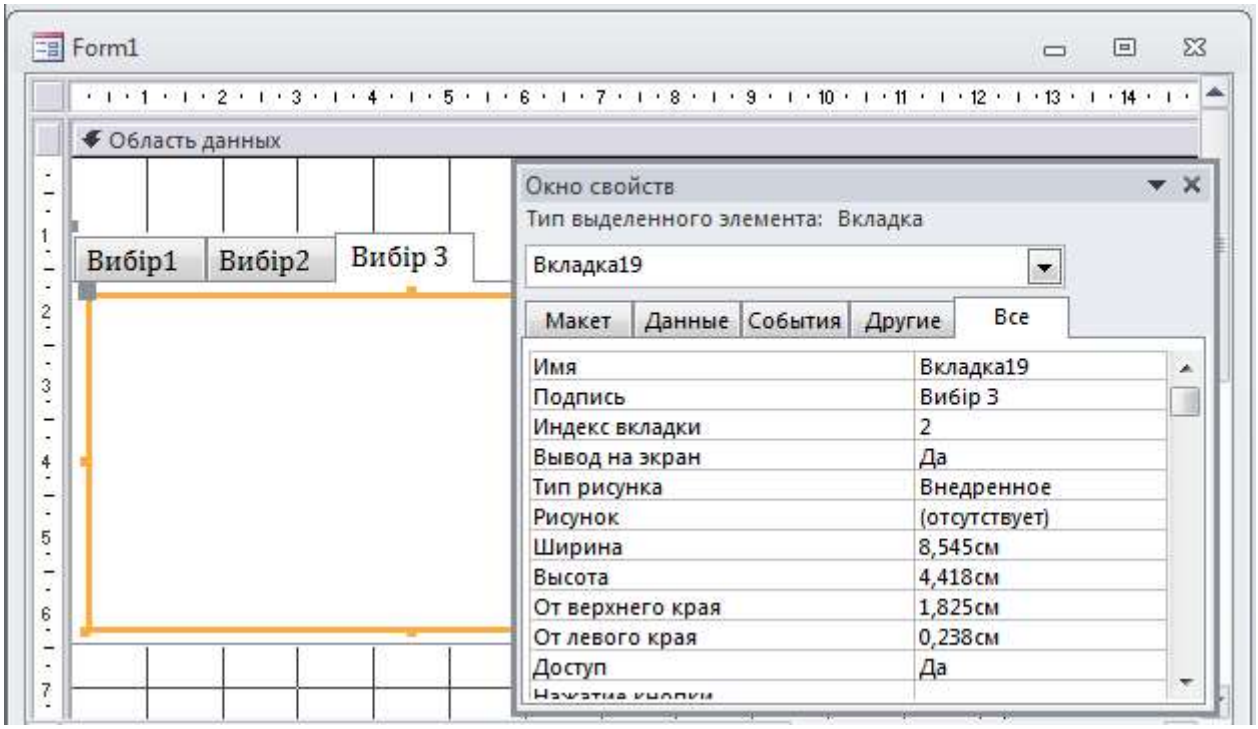



Рис. 9.5 Перейменування Вкладки19

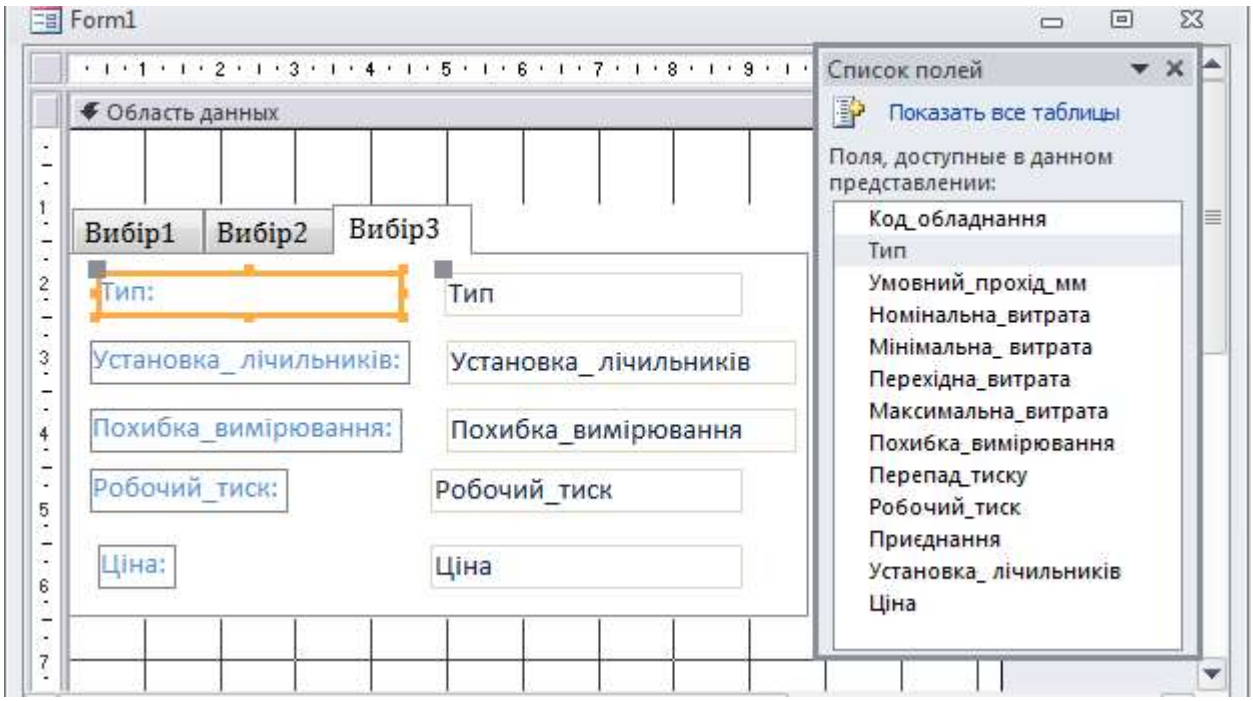



Рис. 9.6. Створення закладки Вибір3 у формі

7. Перетворити всі поля на закладках у Поля зі списком. В полі **Источник строк - Свойства поля** робимо окремо для кожного поля свій запит.

Для поля **Тип** – рис. 9.7.1, для поля **Умовний прохід від** - рис. 9.7.2, для поля **Умовний прохід до** - рис. 9.7.3, для поля **Перепад тиску до** - рис. 9.7.4, для поля **Ціна до** - рис. 9.7.5.

Аналогічно створити запити для всіх полів.

Для кожного поля запис **Данные** - **Свойства поля** зробити порожнім.

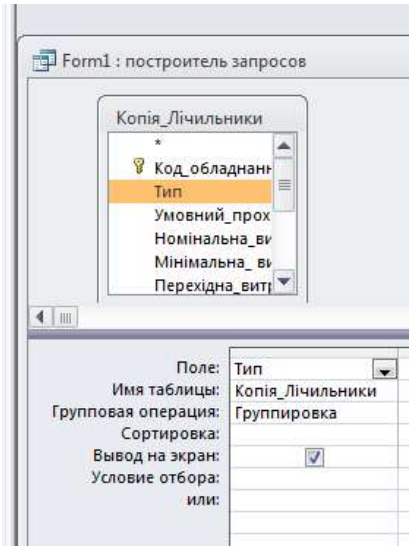



Рис. 9.7.1. Створення запиту для поля **Тип**

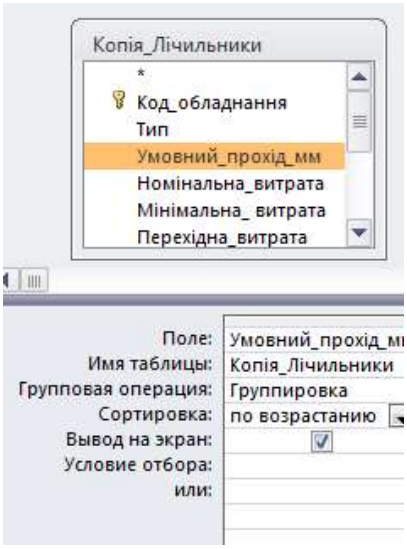



Рис. 9.7.2. Створення запиту для поля **Умовний прохід_мм**

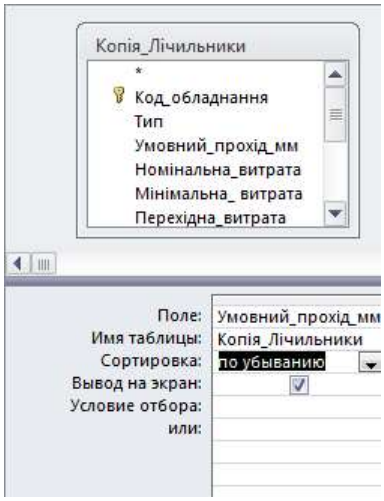



Рис. 9.7.3. Створення запиту для поля **Умовний прохід_мм**

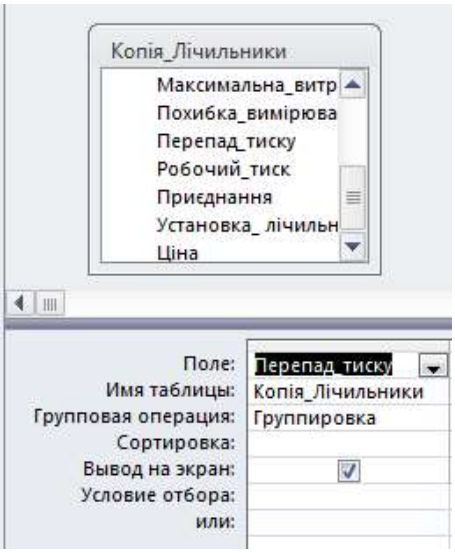



Рис. 9.7.4. Створення запиту для поля **Перепад тиску до**

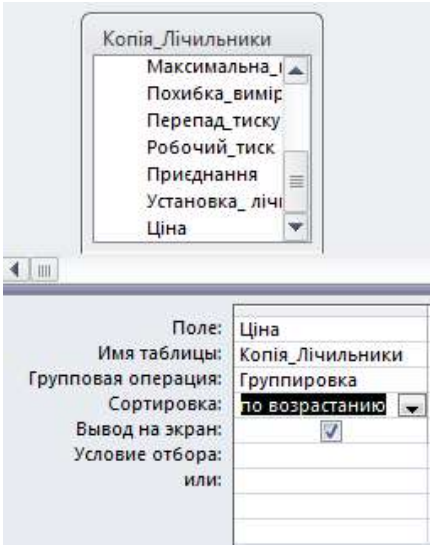
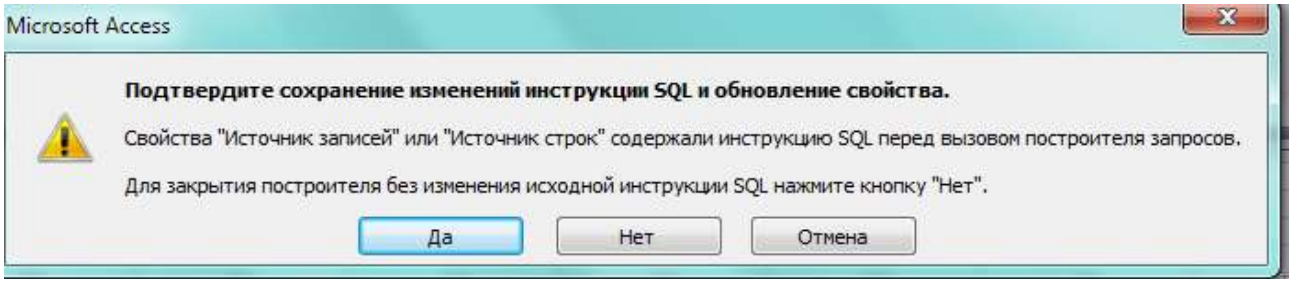



Рис. 9.7.5. Створення запиту для поля **Ціна до**

Після закінчення створення запиту, конструктор необхідно закрити та натиснути кнопку **ДА** у вікні попереджувального повідомлення , зразок якого наведено нижче




Змінити стиль закладок. Для цього потрібно обрати Свойства набора вкладок та в полі **Стиль** значення **Ярлычки** змінити **на НЕТ**, а **Тип фона** вибрати **Прозрачный** (рис. 9.8).

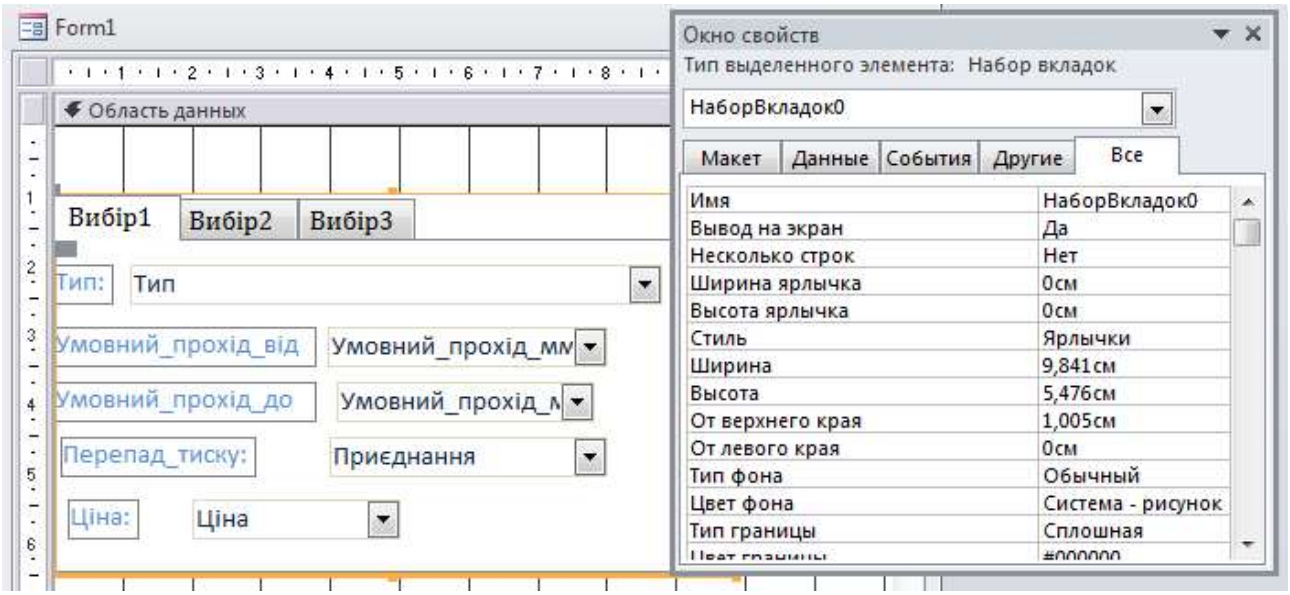



Рис. 9.8. Змінна стилю набора вкладок

8. Зберегти форму з ім'ям **Пошук лічильників**.
9. За допомогою конструктора створити нову форму з ім’ям **Пошук**.
10. Створити закладки натиснувши кнопку  на панелі інструментів.
11. Перейменувати першу закладку, записавши в полі **Подпись - Свойства Вкладки** ім'я закладки **Лічильників**, та зробити на ній 3 кнопки з відповідними підписами, як показано на рис. 9.9.

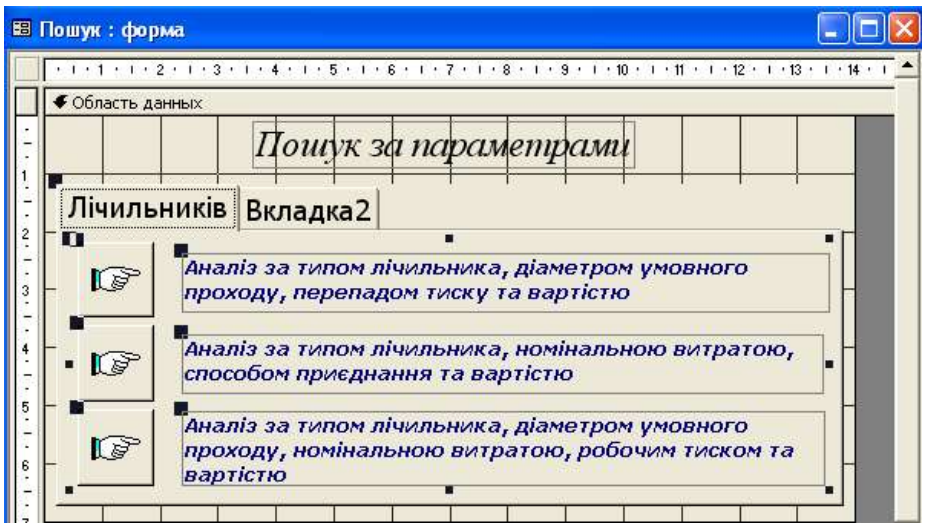



Рис. 9.9. Створення форми **Пошук**

12. Для кожної кнопки створити програму, яка при натисканні цієї кнопки буде відкривати Форму **Пошук лічильників** на необхідній вкладці. Для цього в полі **Свойства – Нажатие кнопки** вибираємо **Программы**. Відкривається вікно **Visual Basic**(рис. 9.10.1) в якому набираємо текст програми.

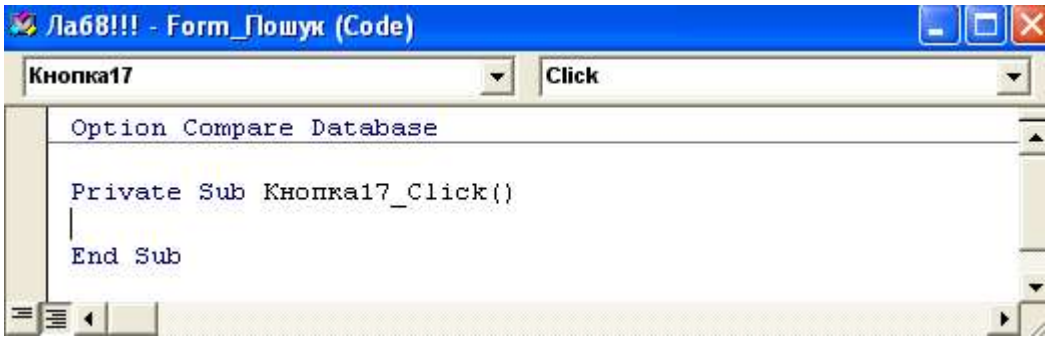



Рис. 9.10.1 Створення програми

Для першої кнопки текст програми буде:

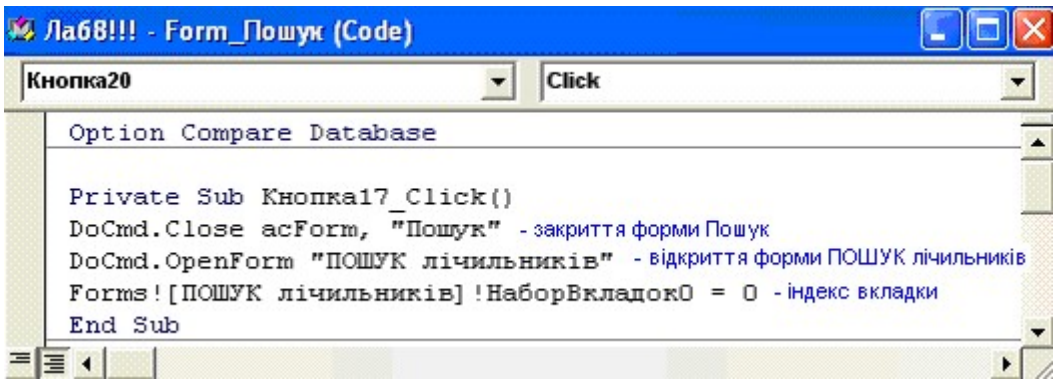



Рис. 9.10.2. Створення програми для першої кнопки

Для другої кнопки текст програми буде:

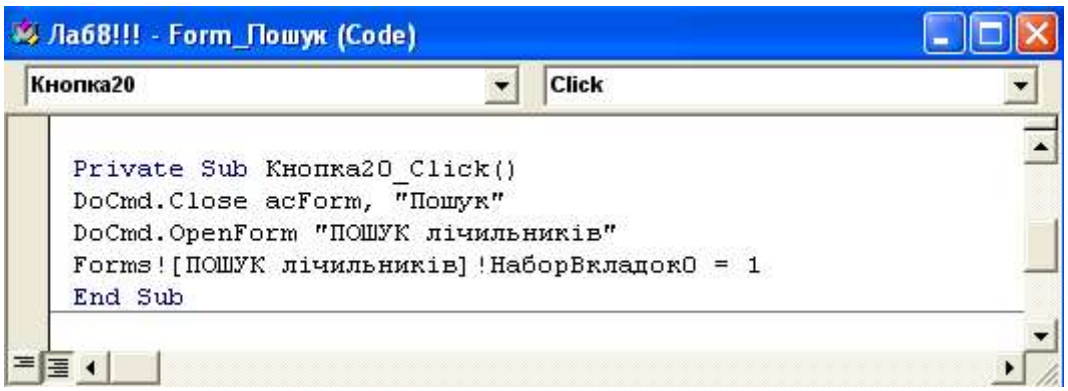



Рис. 9.10.3. Створення програми для другої кнопки

Для третьої кнопки текст програми буде:

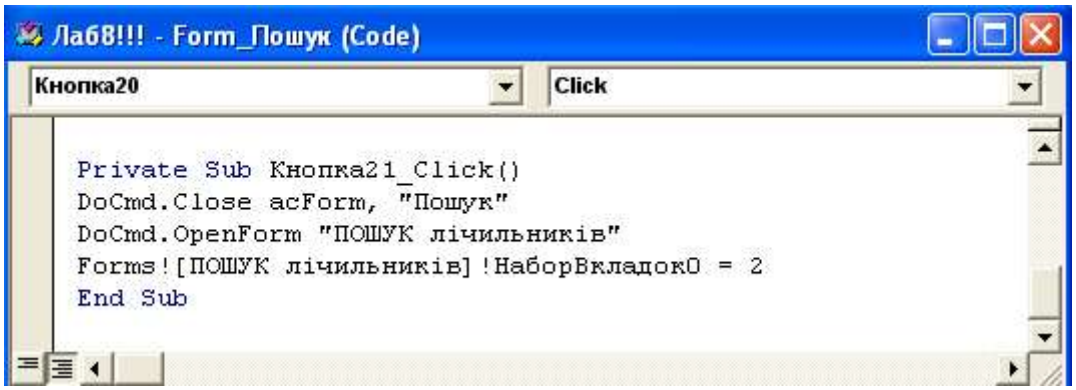



Рис. 9.10.4. Створення програми для третьої кнопки

13. Зберегти форму.

## 9.1. Завдання до комп'ютерного практикуму

1. Приклад 9.1 необхідно виконати всім студентам. А для створення другої закладки обрати таблицю у відповідності до варіанту.
2. Створити таблицю КлапаниN (для варіантів 1,4,8,11), таблицю КраниN (для варіантів 6,10,14), ЗатвориN (для варіантів 3,12,9,15), таблицю ФільтриN (для варіантів 2,5,7,13), використавши запит на створення таблиці. Де N – номер варіанта студента!!!
3. За допомогою конструктора створити нову форму. Вибрати у якості джерела даних таблицю відповідно до варіанту( **КлапаниN або КраниN або ФільтриN** або **ЗатвориN)**. Повторити пункти 2 – 8 для створення цієї форми. Поля для кожної закладки вибрати власноруч. Зберегти форму з ім'ям відповідно до варіанту ( **Пошук КлапанівN або ПошукКранівN або ПошукФільтрівN або ПошукЗатворівN**) .

4. Перейменувати другу закладку, в Формі **ПошукN**, записавши в полі **Подпись - Свойства Вкладки** ім'я закладки **КлапанівN або КранівN або ФільтрівN або ЗатворівN)** та зробити на ній 3 кнопки з відповідними підписами, які відповідають параметрам, по яким буде виконуватися пошук

   Наприклад:

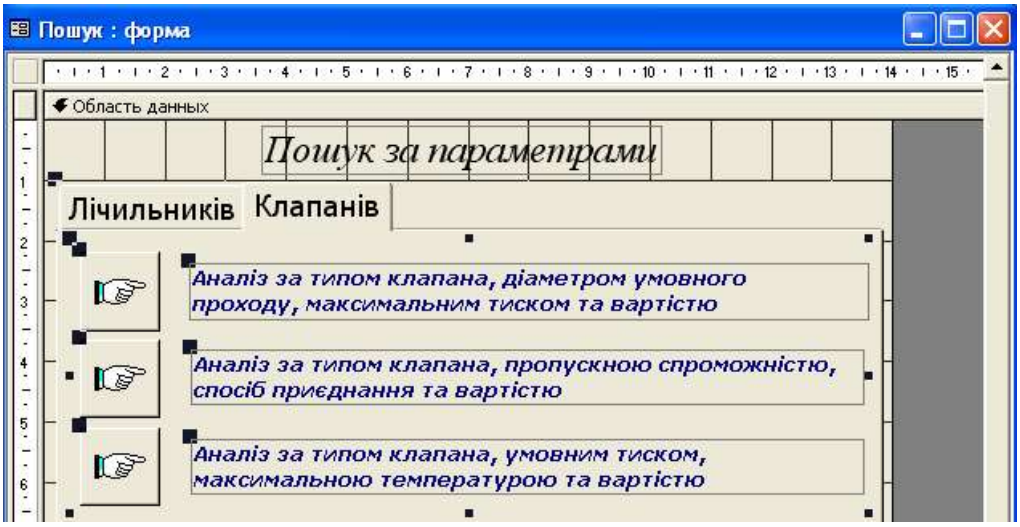



Рис.7.8. Створення заклади на прикладі Клапанів

5. Для кожної кнопки створити програму, яка при натисканні цієї кнопки буде відкривати Форму **Пошук лічильників** на необхідній вкладці.
6. Змінити стиль всіх форм, для цього в полі Полосы прокрутки – Свойства Форми вибирати Нет, Область выделения – Нет, Кнопки перехода – Нет, Разделительные линии – Нет.

## 9.2. Контрольні запитання

1. Для чого використовується об’єкт закладка на формі?
2. Опишіть можливі варіанти створення закладок на формі.
3. Які команди командного процесора використовуються при виконанні комп’ютерного практикуму . Детально опишіть синтаксис та призначення цих команд.
4. Які команди командного процесора DoCmd використовуються для організації роботи кнопок у цьому комп’ютерному практикумі?

# Комп'ютерний практикум № 10. Створення та використання підлеглих форм

*Мета роботи* – набути навичок у створенні підлеглої форми за допомогою конструктора СУБД Access та її використанні для організації роботи з базою даних.

## Теоретичні відомості

При роботі з реляційними базами даних, в яких дані зберігаються в різних таблицях, часто виникає потреба переглядати кілька таблиць або запитів в одній формі. Наприклад, може знадобитися одночасно переглянути дані клієнта з однієї таблиці і відомості про його замовлення з іншої. Підлеглі форми - зручний інструмент для подібних завдань.

Підлеглою формою називають форму, вставлену в іншу форму. Первинна форма називається головною формою, а форма в середині головної форми називається підлеглою. Комбінацію „форма/підлегла форма” часто називають ієрархічною формою або комбінацією „батьківської” та „дочірньої” форм.

Створити підлеглу форму можна одним з наступних способів:

- використати майстер форм для створення нової форми;
- використати майстер підлеглих форм у існуючій формі;
- використати відповідну кнопку на панелі елементів;
- перетягнути форму з вікна бази даних до іншої форми.

Підлеглі форми особливо зручні для виводу даних з таблиці або результатів запитів, зв’язаних з відношенням „один-до-декількох”.

Головна і підлегла в цьому типі форм зв’язані таким чином, що в підлеглій формі виводяться лише ті записи, які зв’язані з поточним записом в головній формі. Наприклад, коли головна форма відображає ім’я та прізвище, підлегла форма відображає товари, які придбав цей покупець.

Не завжди потрібно створювати окремий об'єкт форми для відображення зв'язаних даних. Наприклад, якщо ви працювати з формою в конструкторі або макеті, то перетягнувши в неї таблицю або запит з області навігації призводить в Access до створення підпорядкованої або підлеглої форми з даними в цьому об'єкті. Відображення даних визначається властивістю "Режим за

замовчуванням" відповідного об'єкта. Найчастіше для цього використовується режим таблиці, але для властивості "Режим за замовчуванням" таблиці або запиту можна встановити значення "Одна форма", "Розділена форма" або "Ленточная форма", що забезпечує більшу гнучкість при відображенні пов'язаних даних в формах.

*Примітка:* Зміна властивості Режим за замовчуванням для таблиці або запиту впливає на спосіб їх відображення при відкритті з області навігації або при перегляді в елементі управління підлеглої форми або звіту. Оскільки зміна параметрів відображення таблиці іноді може призвести до плутанини при її відкритті з області навігації, в цій процедурі рекомендується замість таблиці використовувати запит [3].

## 10.1. Вибір бази даних

Для виконання даного комп'ютерного практикуму потрібно скопіювати базу даних **lab_10_stud_pidlegla.accdb** (**Це важливо!!!**). Ця база даних використовується одноразово, лише для вивчення питань організації роботи з підлеглими формами.

В базі даних **lab_10_stud_pidlegla_2021.accdb** створені три форми: Пошук, ПОШУК клапани, ПОШУК лічильників.

- Форма Пошук (рис. 10.1) є головною формою цієї бази даних. На ній розташовані дві закладки, використовуючи які, можна підібрати лічильники чи клапани за заданими параметрами.
- Форми ПОШУК лічильників та ПОШУК клапани (рис.10.2) представляють собою набір закладок (по три на кожній). При натисканні однієї з кнопок на закладках форми Пошук відкривається нова закладка для вибору обладнання з переліченими характеристиками.

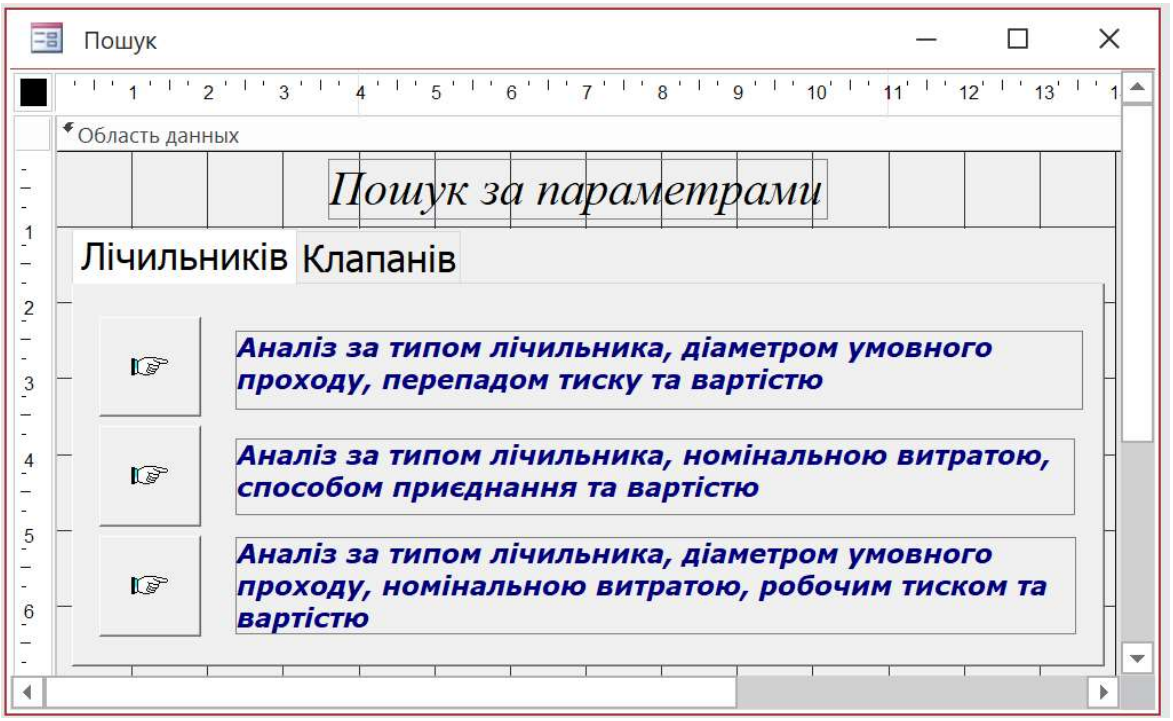



Рис. 10.1. Головна форма бази даних lab_10_stud_pidlegla_2021.accdb

Значення параметрів відповідних характеристик знаходяться у полях ці списками, що значно полегшує процес пошуку.

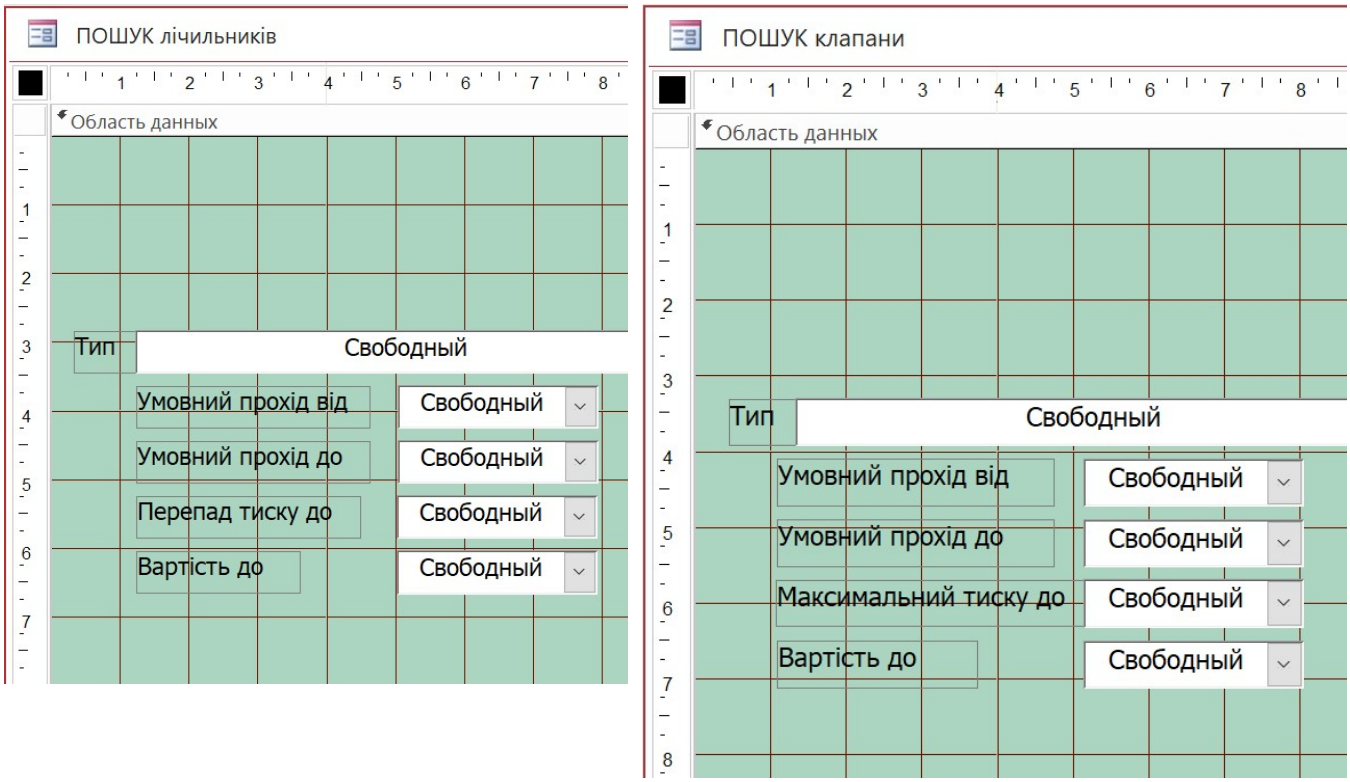



Рис. 10.2. Вигляд заданих форм ПОШУК лічильників та ПОШУК клапани бази даних lab_10_stud_pidlegla.accdb

Пошук обладнання за обраними значеннями параметрів реалізовано за допомогою запитів. Ці запити ПОШУК ЛІЧИЛЬНИКІВ та ПОШУК КЛАПАНІВ також знаходяться в базі даних lab_10_stud_pidlegla.accdb.

```
SELECT [Лічильники+].*
FROM [Лічильники+]
WHERE Switch(forms![ПОШУК лічильників]!НаборВкладок0=0,IIf(Forms![ПОШУК
лічильників]!Тип Is Null,"*",([Лічильники+].Тип)=Forms![ПОШУК лічильників]!Тип) And
(IIf(Forms![ПОШУК лічильників]!Поле5 Is Null,"*",Forms![ПОШУК
лічильників]!Поле5<=([Лічильники+].[Умовний прохід, мм]))) And (IIf(Forms![ПОШУК
лічильників]!ПолеСоСписком7 Is Null,"*",([Лічильники+].[Умовний прохід,
мм])<=Forms![ПОШУК лічильників]!ПолеСоСписком7)) And (IIf(Forms![ПОШУК
лічильників]!Поле9 Is Null,"*",([Лічильники+].Перепад_тиску)<=Forms![ПОШУК
лічильників]!Поле9)) And (IIf(Forms![ПОШУК лічильників]!Поле11 Is
Null,"*",([Лічильники+].Ціна)<=Forms![ПОШУК лічильників]!Поле11)),forms![ПОШУК
лічильників]!НаборВкладок0=1,IIf(Forms![ПОШУК лічильників]!ПолеСоСписком34 Is
Null,"*",([Лічильники+].Тип)=Forms![ПОШУК лічильників]!ПолеСоСписком34) And
(IIf(Forms![ПОШУК лічильників]!ПолеСоСписком36 Is Null,"*",Forms![ПОШУК
лічильників]!ПолеСоСписком36<=([Лічильники+].Номінальна_витрата))) And
(IIf(Forms![ПОШУК лічильників]!ПолеСоСписком38 Is
Null,"*",([Лічильники+].Номінальна_витрата)<=Forms![ПОШУК
лічильників]!ПолеСоСписком38)) And (IIf(Forms![ПОШУК лічильників]!ПолеСоСписком40
Is Null,"*",([Лічильники+].Приєднання)=Forms![ПОШУК
лічильників]!ПолеСоСписком40)) And (IIf(Forms![ПОШУК лічильників]!ПолеСоСписком42
Is Null,"*",([Лічильники+].Ціна)<=Forms![ПОШУК
лічильників]!ПолеСоСписком42)),forms![ПОШУК
лічильників]!НаборВкладок0=2,IIf(Forms![ПОШУК лічильників]!ПолеСоСписком54 Is
Null,"*",([Лічильники+].Тип)=Forms![ПОШУК лічильників]!ПолеСоСписком54) And
(IIf(Forms![ПОШУК лічильників]!ПолеСоСписком56 Is Null,"*",Forms![ПОШУК
лічильників]!ПолеСоСписком56=([Лічильники+].[Установка_лічильників]))) And
(IIf(Forms![ПОШУК лічильників]!ПолеСоСписком58 Is
Null,"*",([Лічильники+].Похибка_вимірювання)=Forms![ПОШУК
лічильників]!ПолеСоСписком58)) And (IIf(Forms![ПОШУК лічильників]!ПолеСоСписком60
Is Null,"*",([Лічильники+].Робочий_тиск)=Forms![ПОШУК
лічильників]!ПолеСоСписком60)) And (IIf(Forms![ПОШУК лічильників]!ПолеСоСписком62
Is Null,"*",([Лічильники+].Ціна)<=Forms![ПОШУК лічильників]!ПолеСоСписком62)));
```

Рис. 10.3. SQL-код запиту ПОШУК ЛІЧИЛЬНИКІВ

*Примітка.* Основним завданням цього комп’ютерного практикуму є створення підлеглих форм та зв’язування новостворених форм з уже існуючими.

Тому з метою отримання коректно працюючої бази даних бажано всі імена новостворюваних форм копіювати з тексту протоколу комп’ютерного практикуму .

## 10.2. Послідовність створення підлеглої форми

1. За допомогою конструктора створити нову форму „Подчинённая ЛІЧИЛЬНИКІВ” з використанням опцій **Создание/Форма**. Вибрати у якості джерела даних таблицю **Лічильники+** (рис. 10.4).

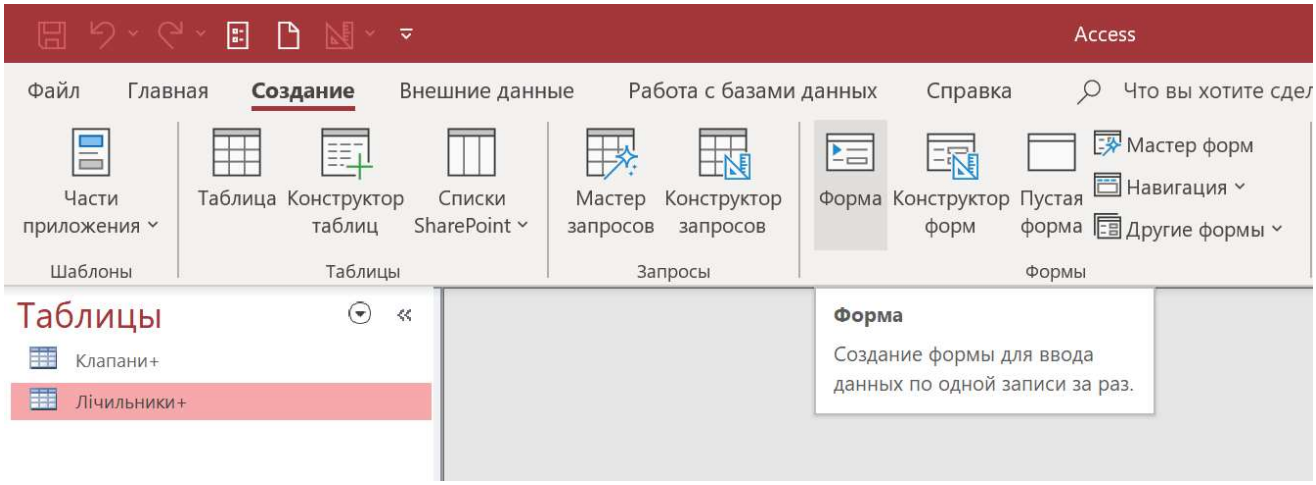



Рис. 10.4. Створення форми „Подчинённая ЛІЧИЛЬНИКІВ” з використанням опцій Создание/Форма

Відкриється форма вигляд якої показано на рис. 10.5. Необхідно змінити з ім’я форми на Подчинённая ЛІЧИЛЬНИКІВ

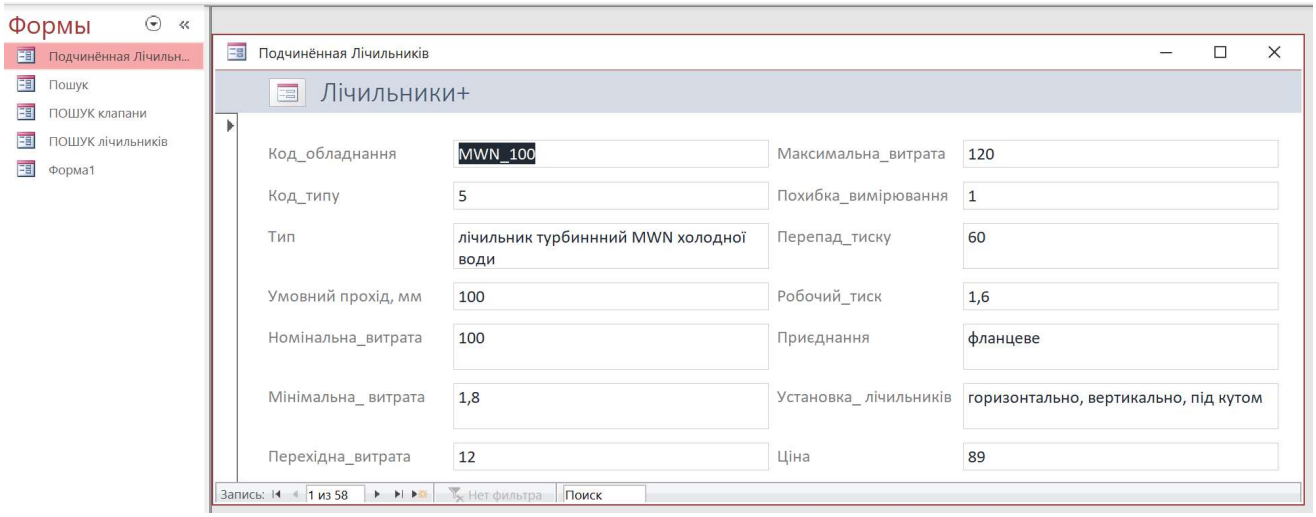



Рис. 10.5. Вигляд створеної форми „Подчинённая ЛІЧИЛЬНИКІВ”

2. Змінити стиль форми, для цього в полі Полосы прокрутки – Свойства Формы вибрати Отсутствуют, Область выбора записей – Нет, Кнопки навигации – Нет, Линия разделения в разделенной форме – Нет, Кнопка оконного меню– Нет. Збережіть цю форму з відредагованими властивостями з ім’ям Подчинённая ЛІЧИЛЬНИКІВ. Та перейдіть до наступного пункту.

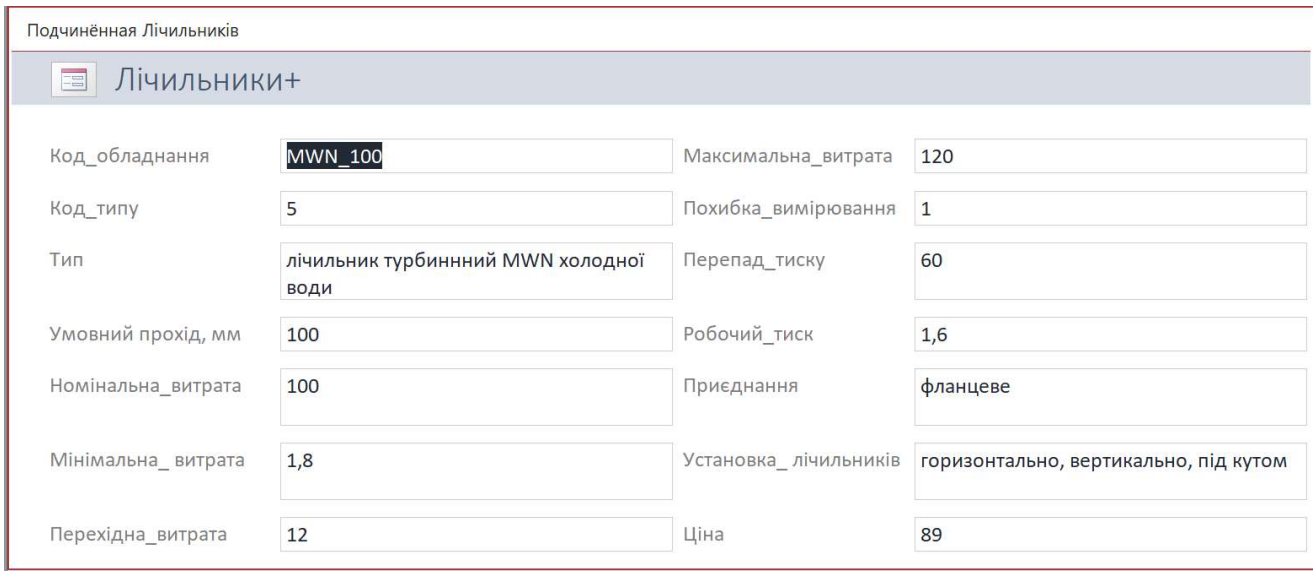



Рис. 10.6. Вигляд форми форми „Подчинённая ЛІЧИЛЬНИКІВ” після зміни стилю

3. За допомогою майстра форм створити ще одну форму. Для якої в якості джерела даних обрати запит **ПОШУК ЛІЧИЛЬНИКІВ**.

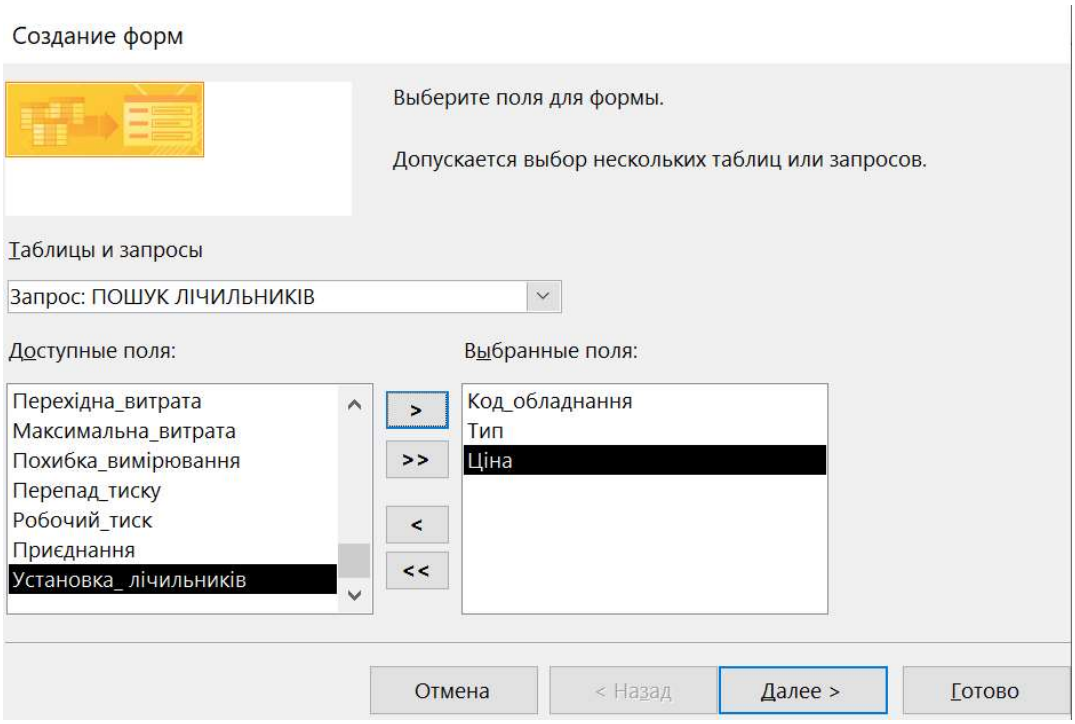



Рис. 10.7. Створення форми „Подчинённая ПОШУК ЛІЧИЛЬНИКІВ”

Для створюваної форми вибрати поля: Код_обладнання, Тип, Ціна. (рис. 10.7) та обрати зовнішній вигляд форми – **ленточный** ( рис. 10.8).

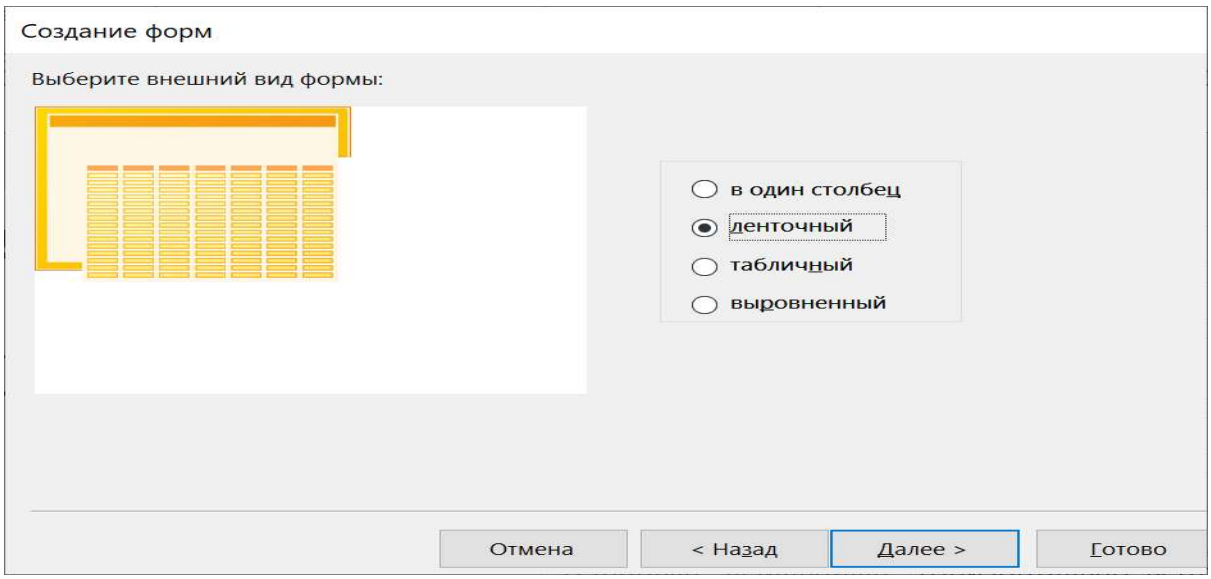



Рис. 10.8. Вибір стилю форми „Подчинённая ПОШУК ЛІЧИЛЬНИКІВ”

Надати ім‘я створеній формі: „**Подчинённая ПОШУК ЛІЧИЛЬНИКІВ**” та обрати опцію **Изменить макет формы.** Бажано це ім‘я скопіювати з тексту протоколу комп’ютерного практикуму для коректної роботи попередньо створеного розробниками багатоваріантного запиту (рис. 10.3).

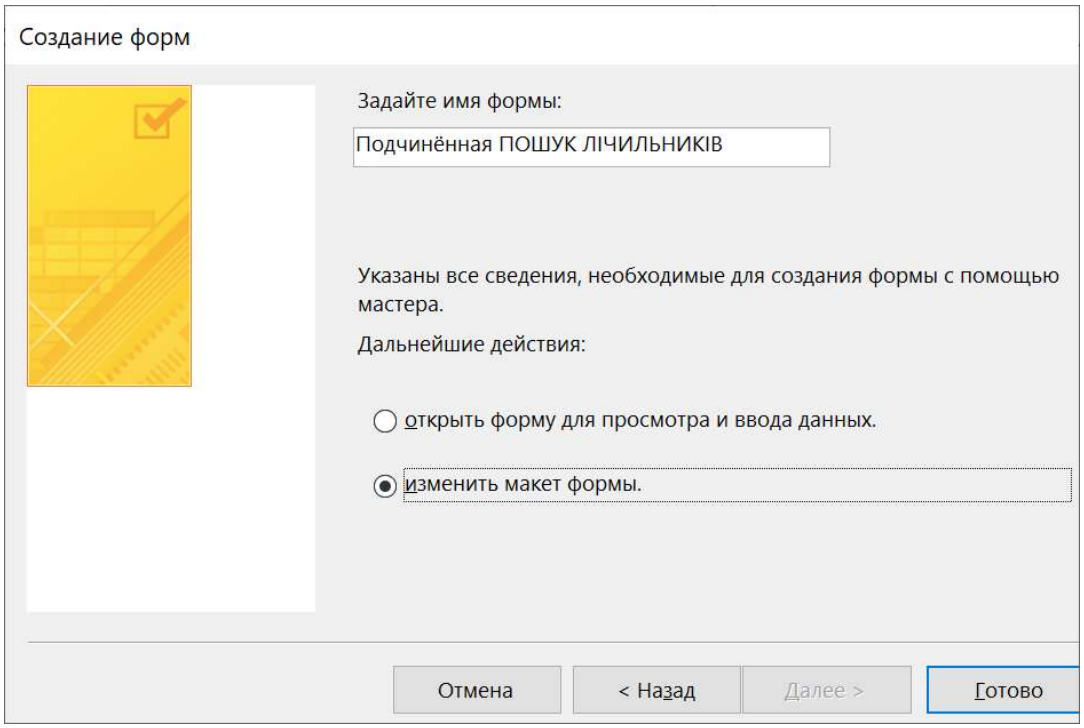



Рис. 10.9. Етап створення форми „Подчинённая ПОШУК ЛІЧИЛЬНИКІВ”

!!! Обов‘зково назвати форму: „Подчинённая ПОШУК ЛІЧИЛЬНИКІВ”.

Аналізуючи SQL-код , можна побачити, що для вибору обладнання з відповідними характеристиками використовуються поля із списками , які зв'язані з попередньо створеними елементами керування форм.

**4.** Зберегти створену форму, але що **дуже важливо**!!! - не закривати її.

!!! Не намагайтесь переглянути дані у вигляді форми.

5. Перейдемо до створення підлеглої форми у формі **Подчинённая ПОШУК ЛІЧИЛЬНИКІВ.** Для цього необхідно натиснути на панелі інструментів піктограму **Подчиненные формы** і намалювати у області **Примечания формы** створеної форми **Подчинённая ПОШУК ЛІЧИЛЬНИКІВ** ділянку для підлеглої форми (рис. 10.10).

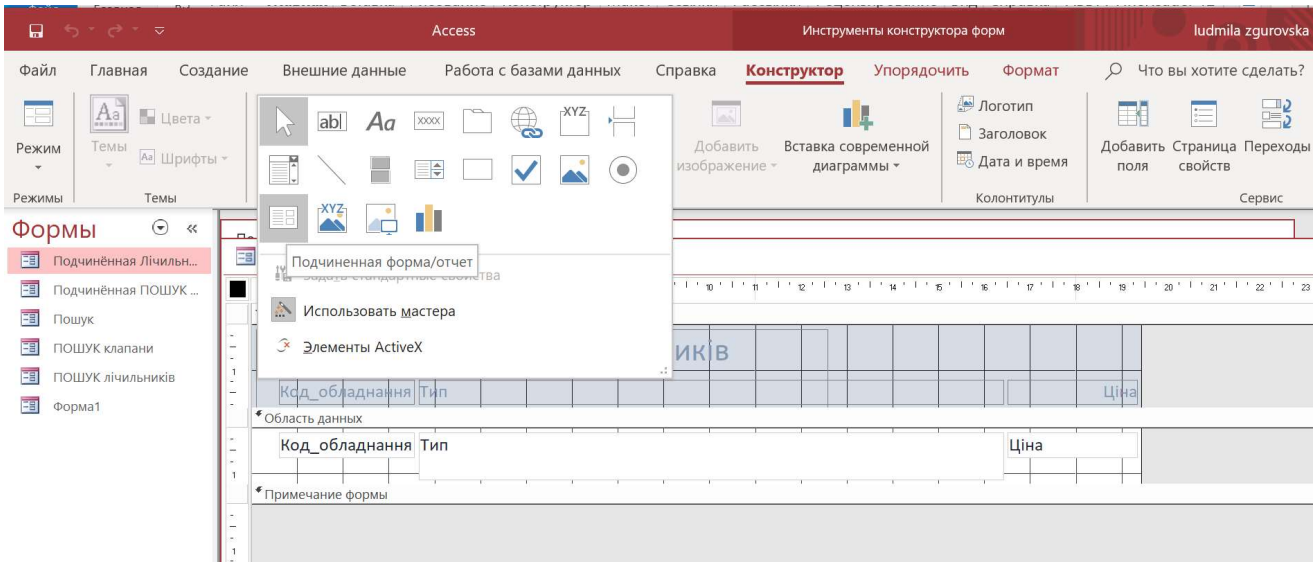



Рис. 10.10. Вибір піктограми **Подчиненная форма/отчет** з панелі елементів керування

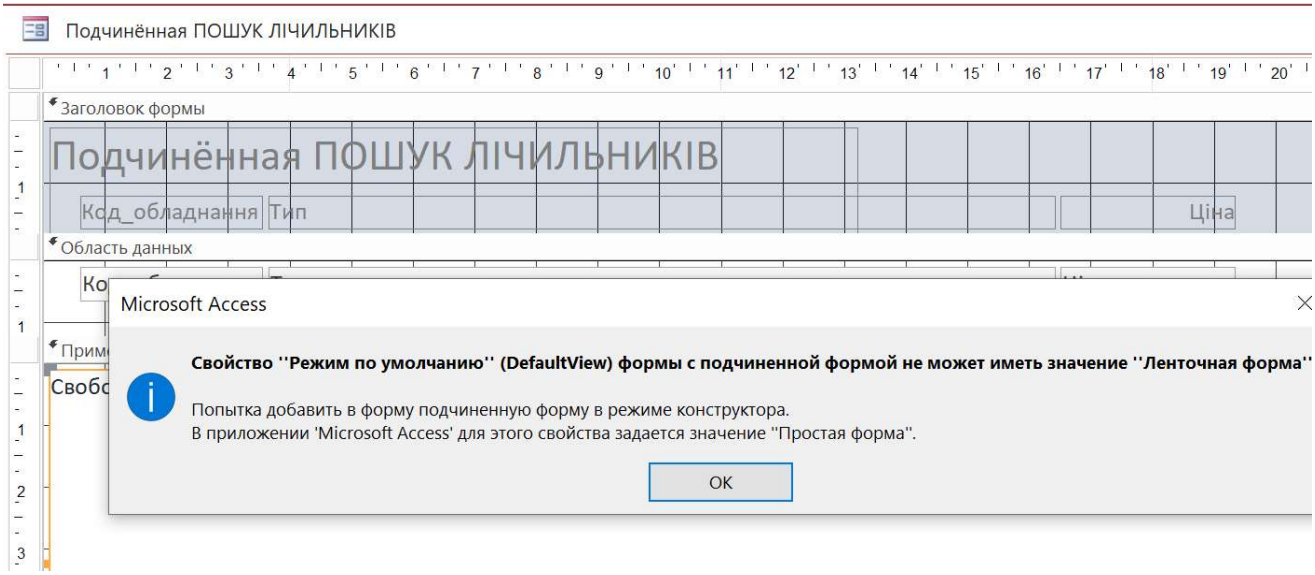



Рис. 10.11. Інформаційне вікно при створенні підлеглої форми

При створенні підлеглої форми можливо отримання інформації (рис. 10.11). Інформацію необхідно прийняти до відома, натиснути кнопку ОК та перейти для вибору джерела даних створюваної підлеглої форми. Вибір джерела даних підлеглої форми може виконуватися з використанням одного з двох варіантів : з активованою опцією Использовать мастера (рис. 10.15) та без активованої опції Использовать мастера (рис. 10.12).

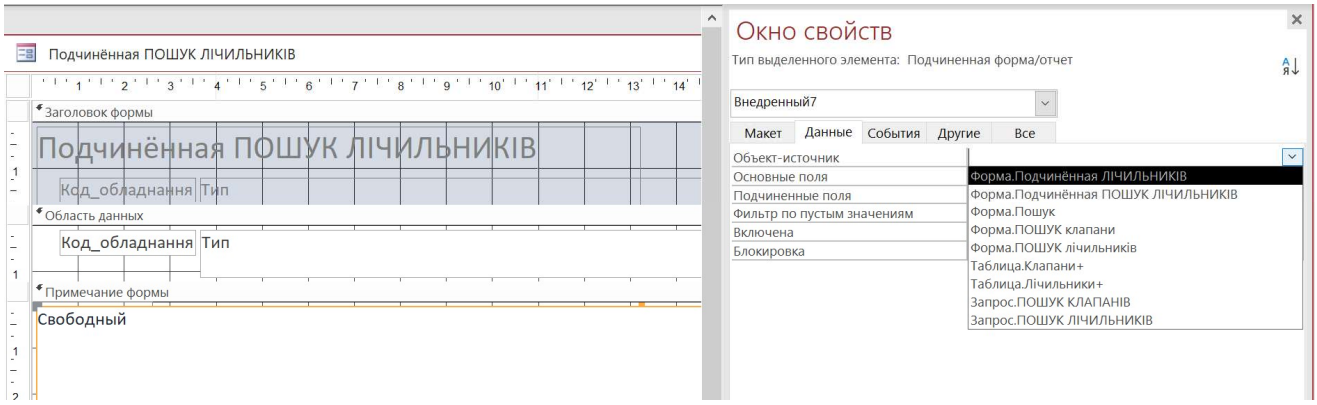



Рис. 10.12. Додавання підлеглої форми без активованої опції **Использовать мастера**

Після цього обрати вже існуючу форму ”**Подчинённая ЛІЧИЛЬНИКІВ**”

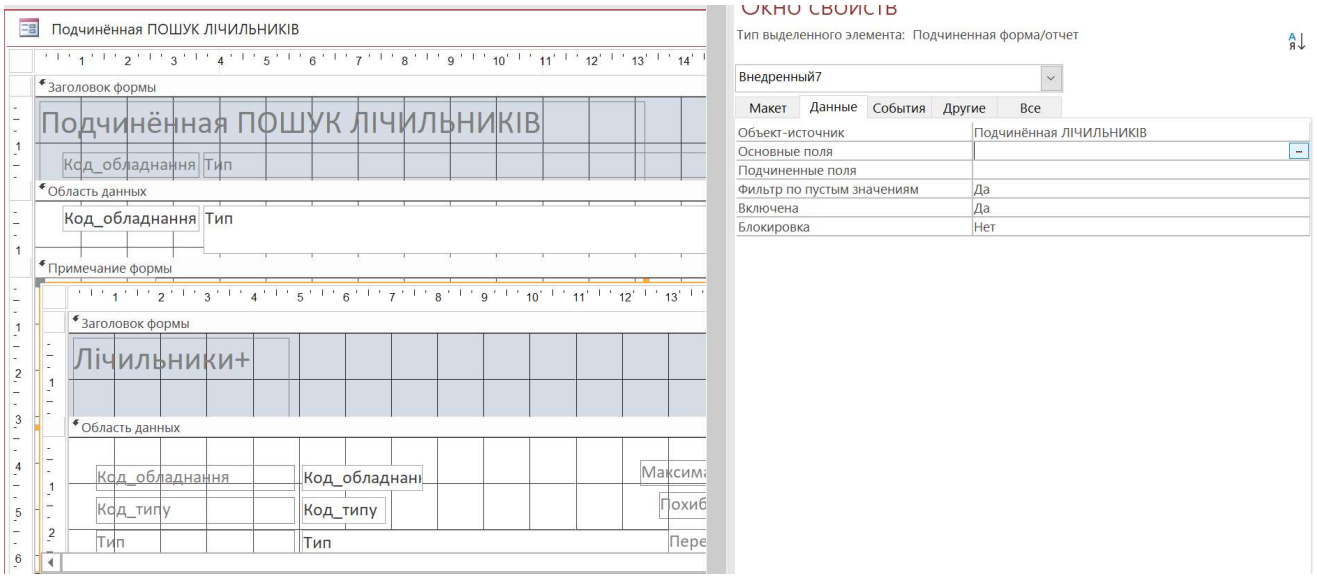



Рис. 10.13. Вибір полів для зв’язку основної форми з підлеглою формою

Встановити курсор в поле **Подчиненные поля** та у вікні, що відкриється, виставити параметри, як показано на рис. 10.14. Тобто встановити зв'язок між двома формами по полю Код_обладнання.

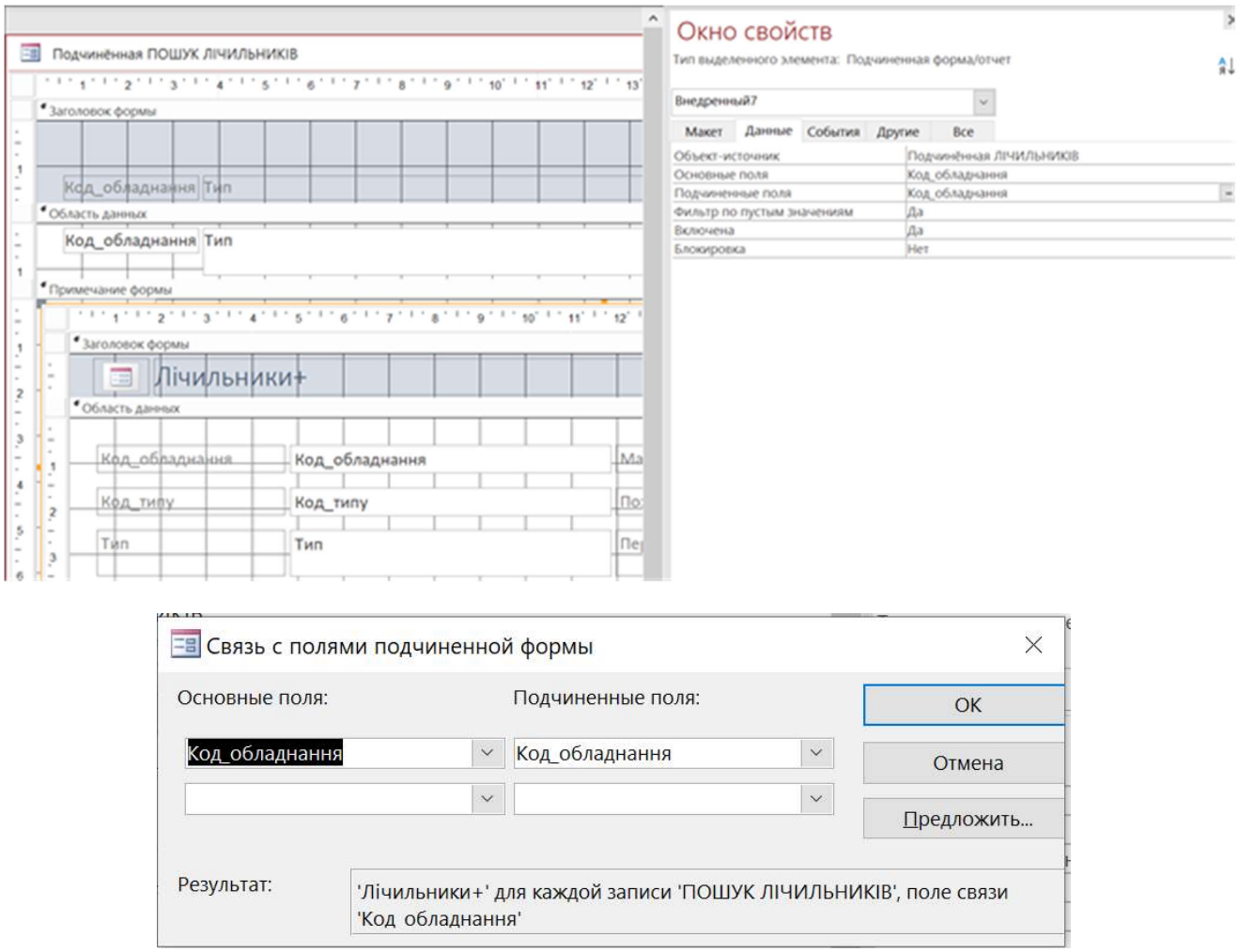



Рис. 10.14 Вибір полів для зв’язування форм без використання опції **Использовать мастера** при створенні підлеглих форм

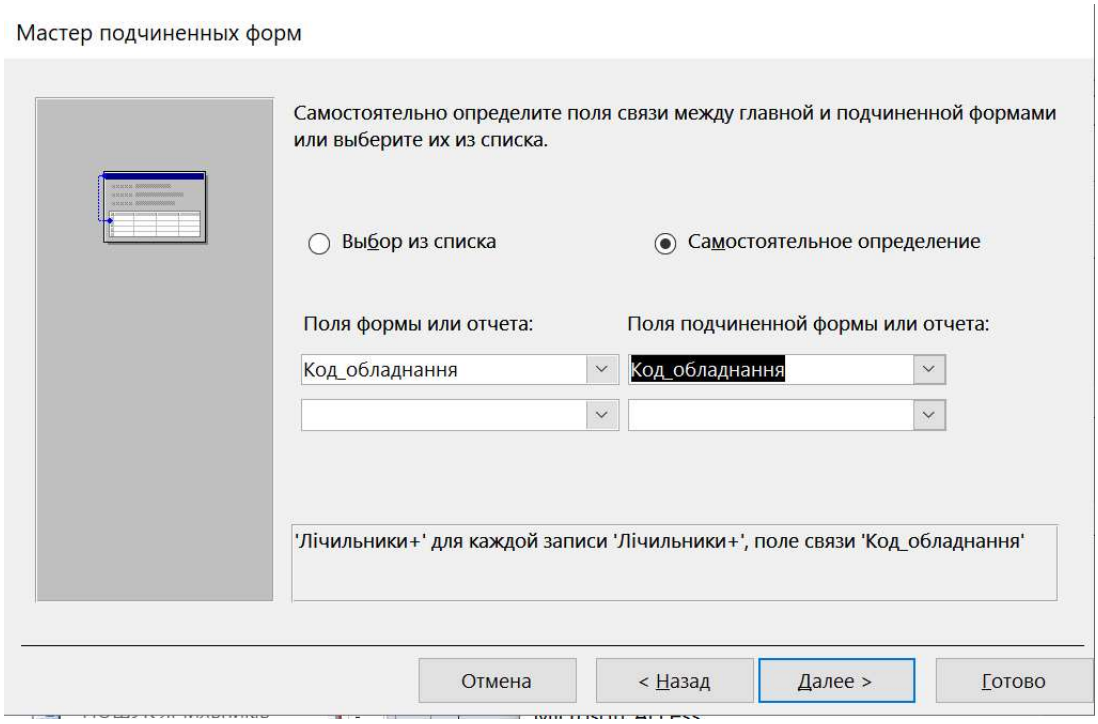



Рис. 10.15. Вибір полів для зв’язування форм з використанням Мастера подчиненных форм

Зберегти створені параметри підлеглої форми, натиснувши **Готово** та після цього зберегти саму форму.

**6.** В режимі конструктора змінити підпис підлеглої форми на „Характеристики лічильника” (рис.10.16). Змінити стиль форми, для цього в полі **Область выделения – Свойства Форми** вибрати **Нет.**

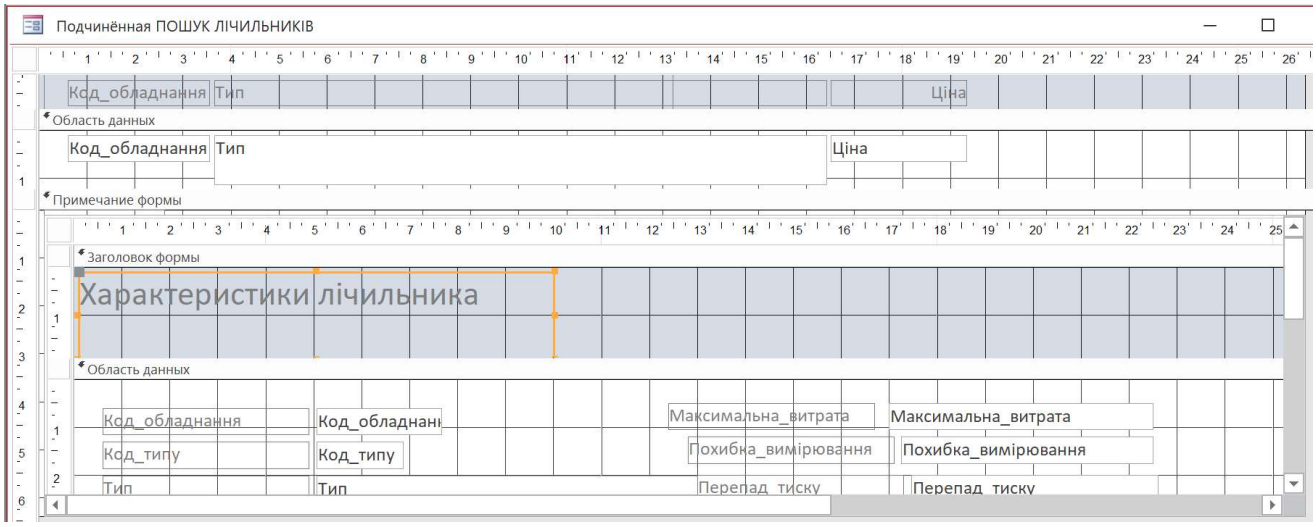



Рис. 10.16. Зміна підпису підлеглої форми

7. Зберегти та закрити форму.

8. Відкрити за допомогою конструктора форму **ПОШУК лічильників**

Натиснути на панелі інструментів на кнопку **Подчиненные формы** і намалювати ділянку для підлеглої форми (рис. 10.17).

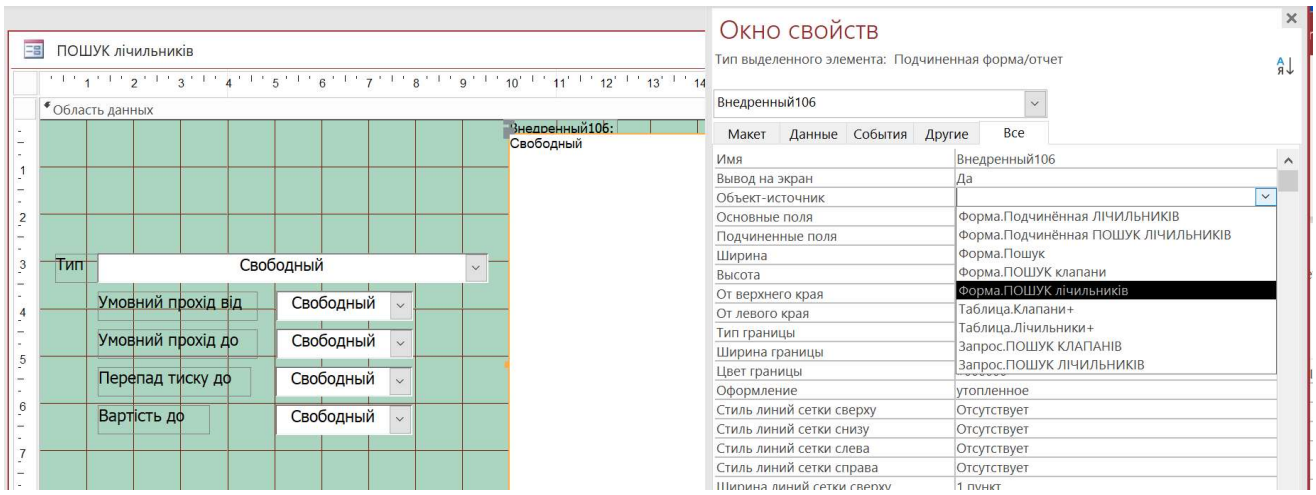



Рис. 10.17. Додавання підлеглої форми у форму **ПОШУК лічильників**

Вибрати вже існуючу форму ”Подчинённая ПОШУК ЛІЧИЛЬНИКІВ” (рис. 10.18).

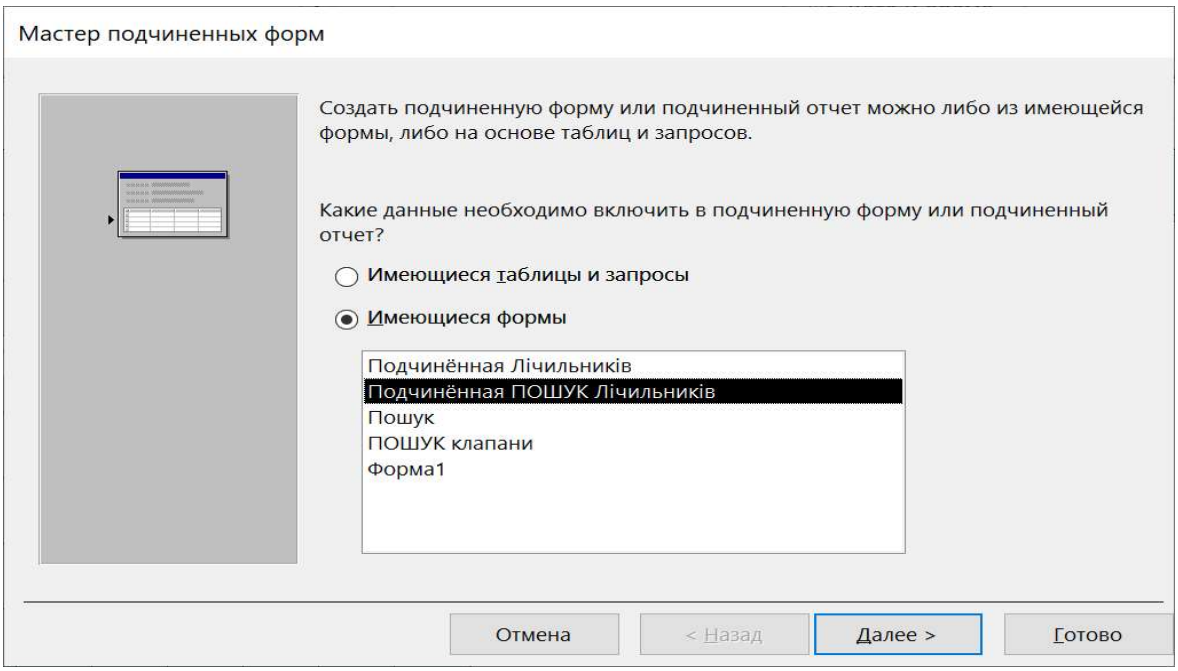



Рис. 10.18. Вибір підлеглої форми

Задати назву підлеглій формі та натиснути **Готово**.

9. Змінити підпис підлеглої форми на „Знайдено” (рис. 10.19).

Змінити стиль тексту, для цього в полі Размер шрифта– Свойства надписи Знайдено вибирати 18, Цвет текста - 16711680.

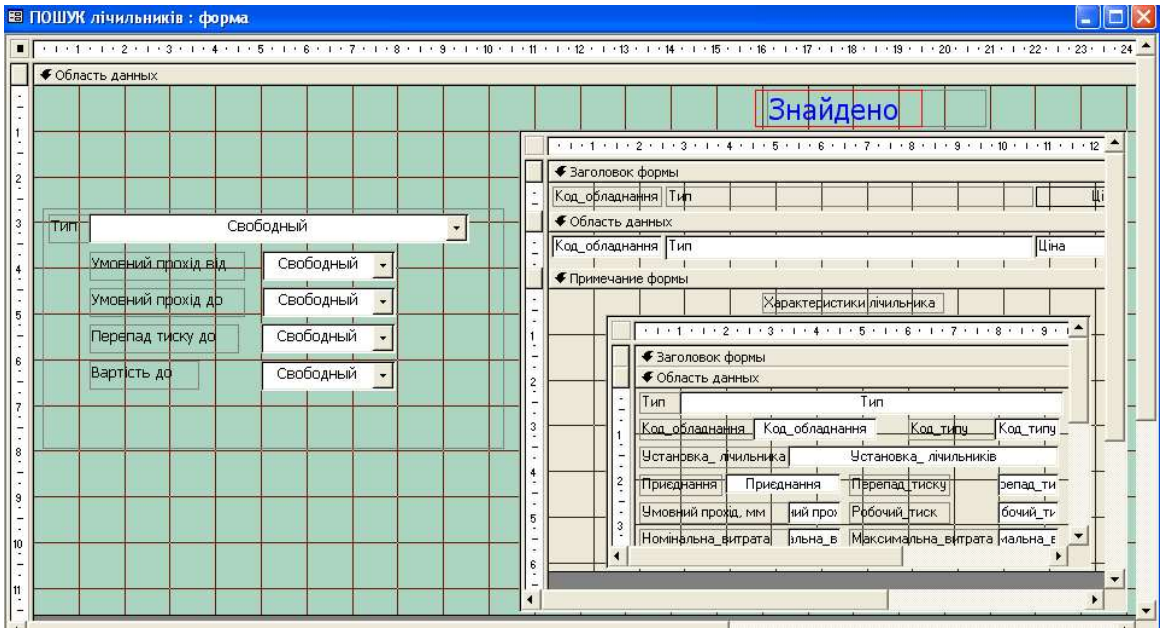



Рис. 10.19. Зміна підпису підлеглої форми

10. Зберегти та закрити форму.

11. Відкрити форму **Пошук.**

Натиснути на будь-яку кнопку на вкладці **Лічильників.** Відкриється форма **ПОШУК лічильників.**

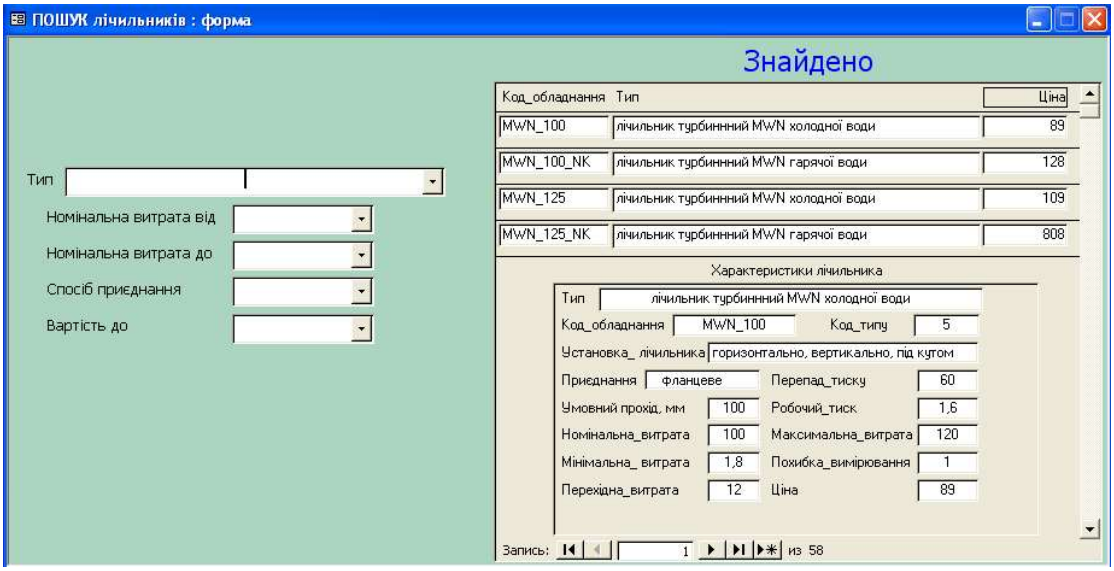



Рис. 10.20. Форма **ПОШУК лічильників**

Тепер необхідно перевірити наступне :

1.Чи відображається на формі підлегла форма Подчинённая ПОШУК ЛІЧИЛЬНИКІВ та Подчинённая ЛІЧИЛЬНИКІВ?

2. Чи правильно працює пошук та перегляд даних?

Якщо щось не працює, значить була допущена помилка!!! Повторіть всі попередні пункти заново.

12. Зробити форму **Пошук** стартовою, щоб при відкритті бази даних автоматично відкривалась форма **Пошук**. Для цього необхідно

Натиснути **Файл → Параметры,** після чого відкриється вікно з параметрами (рис. 10.21) та обрати ім’я форми - Пошук (рис. 10.22).

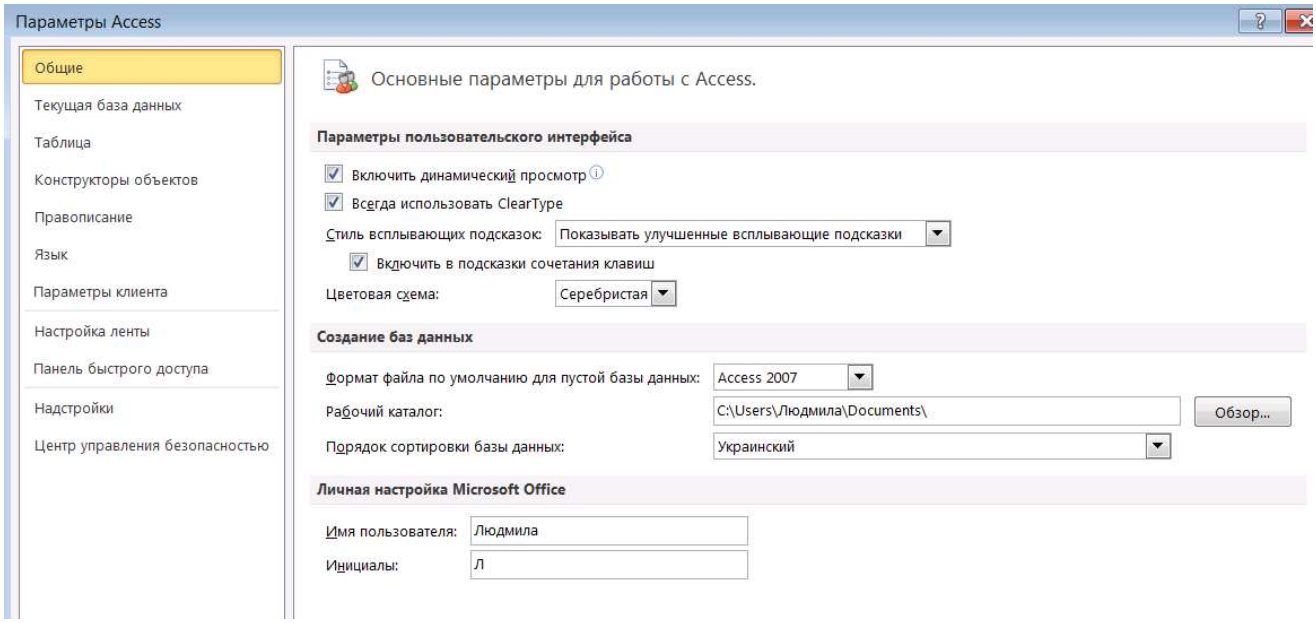



Рис. 10.21. Вибір параметрів застосування

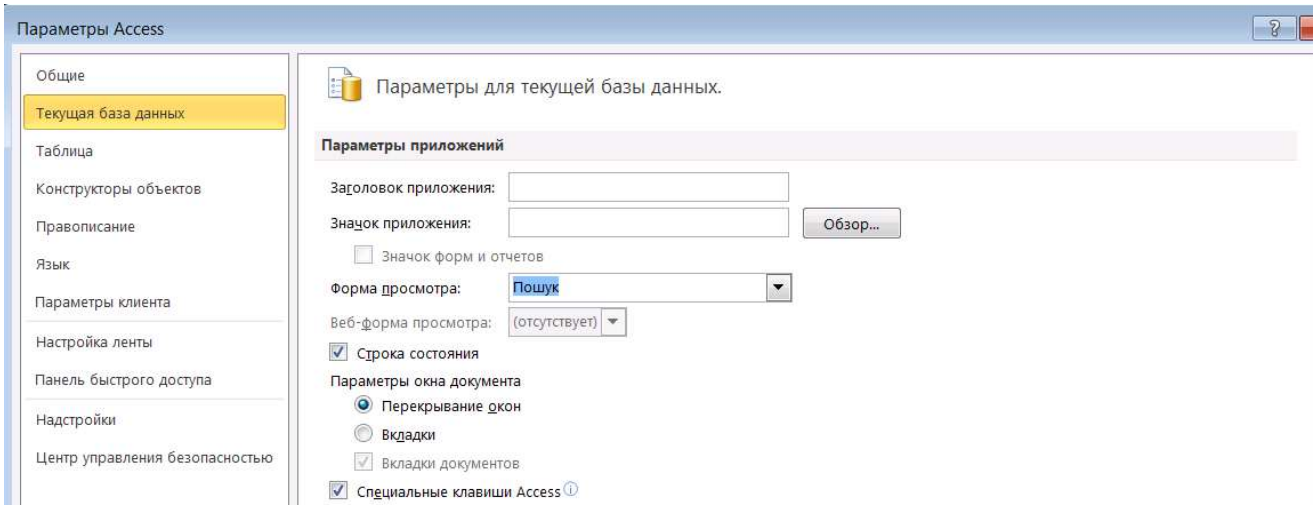



Рис. 10.22. Налагодження параметрів запуску застосування

## 10.3. Завдання до комп'ютерного практикуму

1. Виконати всі завдання по створенню об’єктів для пошуку лічильників за обраними значеннями параметрів характеристик**.** Для цього потрібно виконати пункти 1-10.
2. Виконати всі завдання по створенню об’єктів для пошуку лічильників за обраними значеннями параметрів характеристик. Для цього потрібно виконати пункти 1-10. В першому пункті необхідно створити форму для таблиці Клапани+.

!!! У третьому пункті обов‘язково назвати форму: „Подчинённая ПОШУК КЛАПАНІВ”

Перевірити роботу пошуку клапанів подібно до пункту 11.

3. У протоколі представити опис створення підлеглих форм для пошуку інформації про клапани.
4. Дати письмові відповіді на контрольні запитання.

## 10.4. Контрольні запитання

1. Які способи створення підлеглих форм використовуються у Access?
2. Які способи створення підлеглих форм доступні на Вашому комп’ютері?
3. Який параметр потрібно обов’язково використати для коректного зв’язування основної і підлеглої форм?
4. Які різновидності форм виведення даних можуть використовуватися при створенні підлеглих форм?
5. Призначення підлеглих форм продемонструйте на прикладах.
6. Чи можна використовувати запит в якості джерела даних підлеглої форми?
7. Яка стандартна функція використовується в запиті для організації підбору інформації, відповідаючій даним набору списочних полів?
8. Для яких таблиць бази даних db_stud_comp_prak.accdb можуть бути створені підлеглі форми ?
9. Які команди реалізовано у Диспетчере кнопочных форм?
10. Чи відчутні переваги використання Диспетчера кнопочных форм для створення стандартної головної кнопочної форми? Сформулюйте ці переваги.

# Комп'ютерний практикум № 11. Звіти в СУБД Access

*Мета роботи* – набути навичок створення звітів за допомогою конструктора та програми майстра в СУБД Microsoft Access.

## Теоретичні відомості

Звітом у базі даних називають об’єкт, у якому дані таблиці чи запиту представляються у вигляді документу зрозумілого іншим людям. Засоби розробки звітів в Access призначені для створення макету звіту, по якому може бути здійснено виведення даних у вигляді вихідного друкованого документу. Ці засоби дозволяють створювати звіт складної структури, який забезпечує зв’язок між даними із багатьох таблиць, їх групування, обчислення підсумкових даних, виведення даних у графічному вигляді.

Звіти Access можна розбити на наступні складові частини або розділи, які виступають в ролі свого роду оболонки для преставлення даних:

**Report Headers/Footers (Заголовок /Примечание отчета)** – заголовок роздруковується у верхній частині першої сторінки і може включати, наприклад, назву звіту, логотип фірми та іншу інформацію загального характеру. Примечание отчета роздруковується у нижній частині останньої сторінки звіту.

**Page Headers/Footers (Верхній і нижній колонтитули)** - верхній колонтитул роздруковується у верхній частині кожної друкованої сторінки звіту( в ньому можна розміщувати будь-який текст), а нижній роздруковується у нижній частині кожної друкованої сторінки звіту( в ньому відображається поточна дата, номер поточної сторінки та загальна кількість сторінок).

**Section Headers/Footers (Колонтитули розділу – Заголовок группы\Примечание группы)** - колонтитули розділу відображаються для кожного групування або розбивання на секції. В одному звіті може вміщуватись до десяти таких розділів, а при друкуванні вони можуть бути представлені стільки разів, скільки потрібно. Колонтитули розділу часто викликають труднощі у розробників при створенні виразів у звіті.

**Detail (Область даних)** - область даних призначена для відображення інформації, що міститься у звіті. Існує лише одна область даних, яка

знаходиться між колонтитулами других розділів в центральній частині вікна конструктора.

Користувачеві доступні два способи створення звіту: за допомогою Майстра Access або вручну, використовуючи набір інструментів Конструктора звітів.

Майстер звітів Report Wizard – ймовірно найкорисніший майстер в Access, і хоча навряд чи він забезпечить розробника звітом, що відповідає всім необхідним вимогам, він здатний виконати 90% всієї роботи для 90% звітів, що створюються. Більш того, задачі, що вирішуються майстром, – це найбільш рутинні задачі, які виникають при створенні звітів.

При використанні Конструктора звітів основним способом представлення даних у звіті є розміщення зв'язаних елементів управління в розділах для управління відображенням даних при друкуванні. Однак, крім того, можна розмістити звіт в будь-якому із перерахованих вище розділів таким чином, що дані будуть представлені як підлеглий звіт. Переглянути створений звіт перед друкуванням можна за допомогою опцій меню Вид - Предварительный просмотр (рис. 11.1,б). Для створення звітів можна використовувати наступні можливості MS Access (рис.11.1,а).

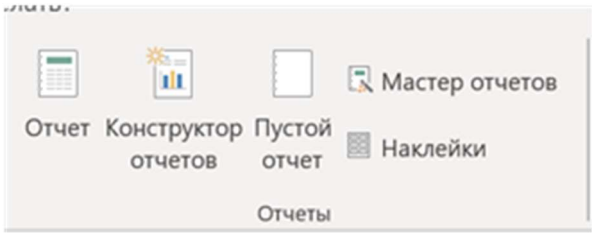



Рис. 11.1,а. Засоби створення звітів MS Access

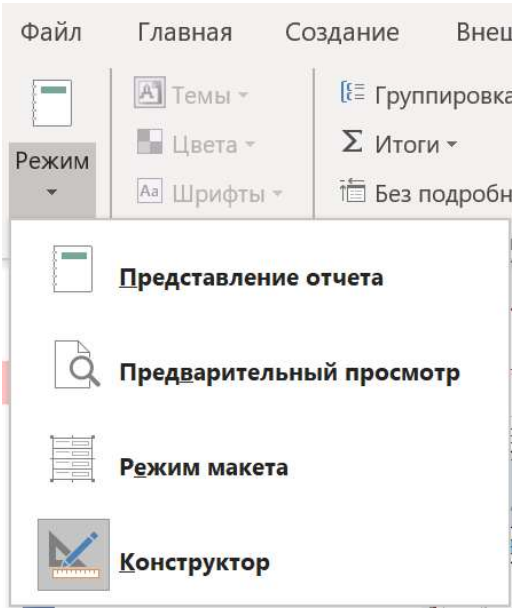



Рис. 11.1,б. Засоби перегляду та модифікації звітів MS Access

**Агрегуючі функції**

| Function | Функція | Опис |
|---|---|---|
| Sum | Сума | Обчислення суми значень |
| Count | Число | Підрахунок кількості значень. Порожні значення, такі як рядок із нульовою довжиною (рядки, що не містять знаків), включаються в результат функції, але значення Null (значення, що позначають відсутні чи невідомі дані) не включаються. |
| Min | Мінімум | Повертає в підсумкове поле найменше значення |
| Max | Максимум | Повертає в підсумкове поле найбільше значення |
| Avg | Середнє | Повертає в підсумкове поле середнє значення |
| Var | Сукупність | Повертає дисперсію значень |

**Приклад 11.1.** Створити запит на підставі таблиць Склад_обладнання, Назва_обладнання, Виробники обладнання. Запит повинен містити такі поля: Код_обладнання, Код_типу, Тип_обладнання, Виробник, Ціна, Кількість_на_складі. Зберегти запит під назвою Склад (рис.11.2).

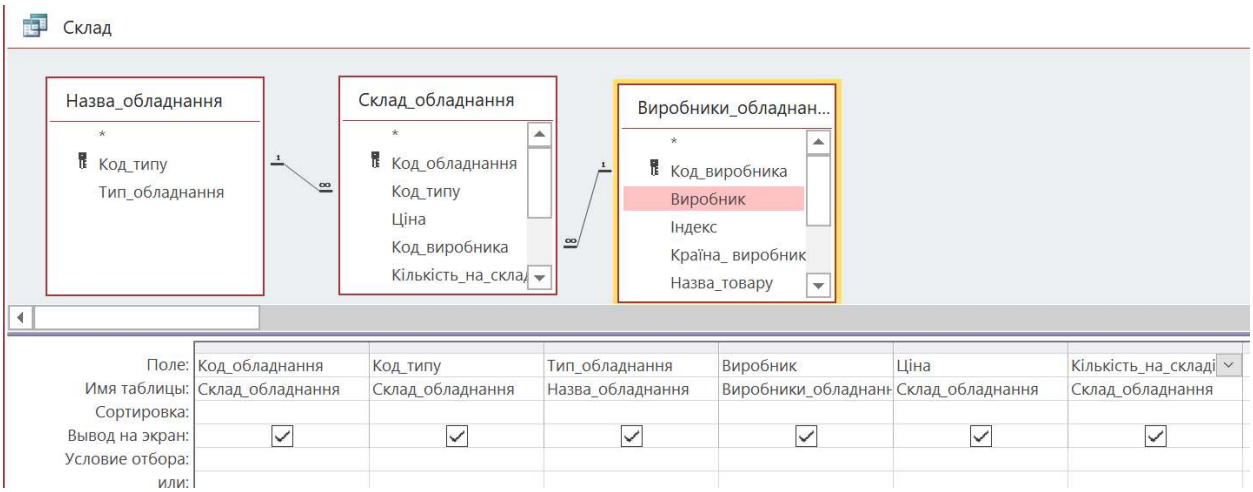



Рис. 11.2. Вікно конструктора запиту **Склад**

1. Для створення **Звіту,** потрібно виділити запит **Склад** та у вікні **Бази даних** обрати опцію меню **Создание,** а потім піктограму **Отчет**.
2. З'являється вікно **Отчет** для запиту **Склад** (рис. 11.3)**.**

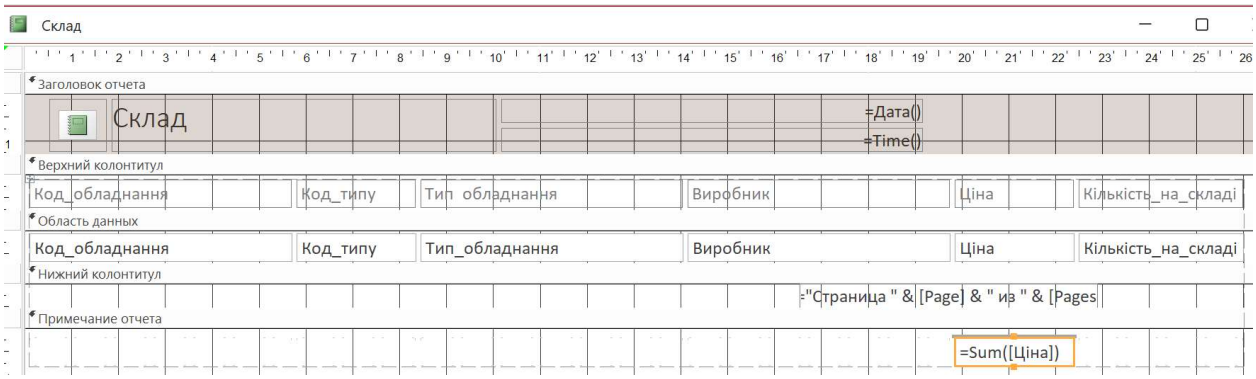
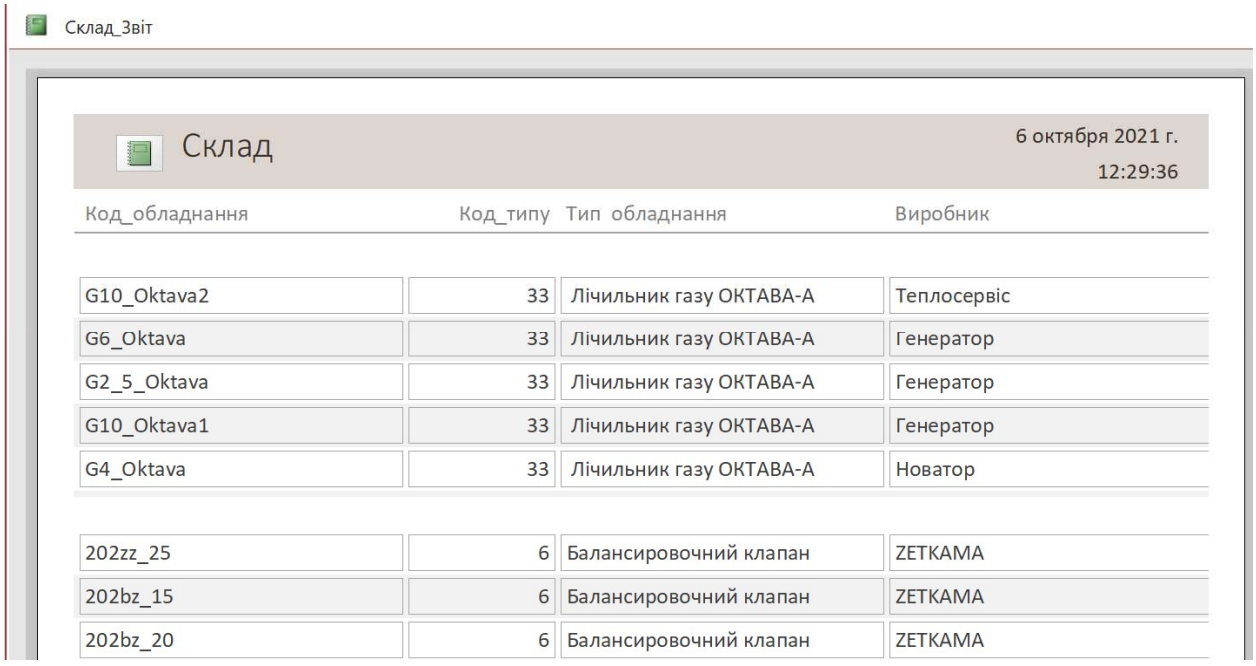



Рис. 11.3. Звіт **Склад_Звіт** в режимі конструктора

3. Виконати групування по полю **Тип_обладнання**. Для цього потрібно виділити поле **Тип_обладнання**, та натиснути кнопку  . Вибрати режим **Предварительный просмотр** . В результаті отримаємо звіт наступного вигляду (рис. 11.4). Зберегти створений звіт з ім'ям Склад_Звіт_Прізвище_студента.




| Код_обладнання | Код_типу | Тип обладнання | Виробник |
|---|---|---|---|
| G10_Oktava2 | 33 | Лічильник газу ОКТАВА-А | Теплосервіс |
| G6_Oktava | 33 | Лічильник газу ОКТАВА-А | Генератор |
| G2_5_Oktava | 33 | Лічильник газу ОКТАВА-А | Генератор |
| G10_Oktava1 | 33 | Лічильник газу ОКТАВА-А | Генератор |
| G4_Oktava | 33 | Лічильник газу ОКТАВА-А | Новатор |
| 202zz_25 | 6 | Балансировочний клапан | ZETKAMA |
| 202bz_15 | 6 | Балансировочний клапан | ZETKAMA |
| 202bz_20 | 6 | Балансировочний клапан | ZETKAMA |

Рис. 11.4. Звіт Склад_Звіт в режимі Предварительный просмотр

**Приклад 11.2.** Створити звіт з використанням Мастер отчетов (ступенчатый) для запиту Склад, виконати групування по полю Виробник, впорядкування по полю Ціна. Скрін результату записати до протоколу.

Розглянемо режими створення звіту з використанням Мастер отчетов. Цей режим надає можливості створення звіту як для таблиць, вибраних в якості джерела даних так і для запитів (рис. 11.5).

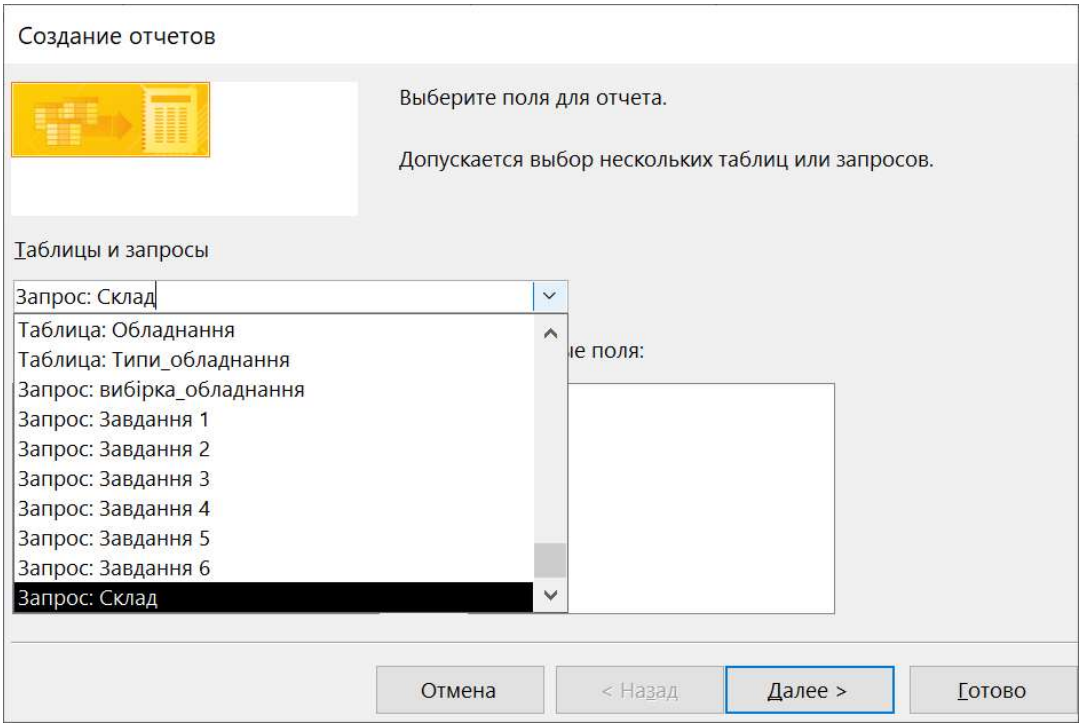



Рис. 11.5. Вибір джерела даних для створення звіту

Також надаються можливості групування та сортування інформації (рис. 11.6, 11.7) та вибору макету звіту: ступенчатый (в столбец), блок (табличный), структура (выровненный) (рис. 11.8,а, рис. 11.8,б). Майстром звітів надається можливість обрати орієнтацію звіту: Книжная або Альбомная.

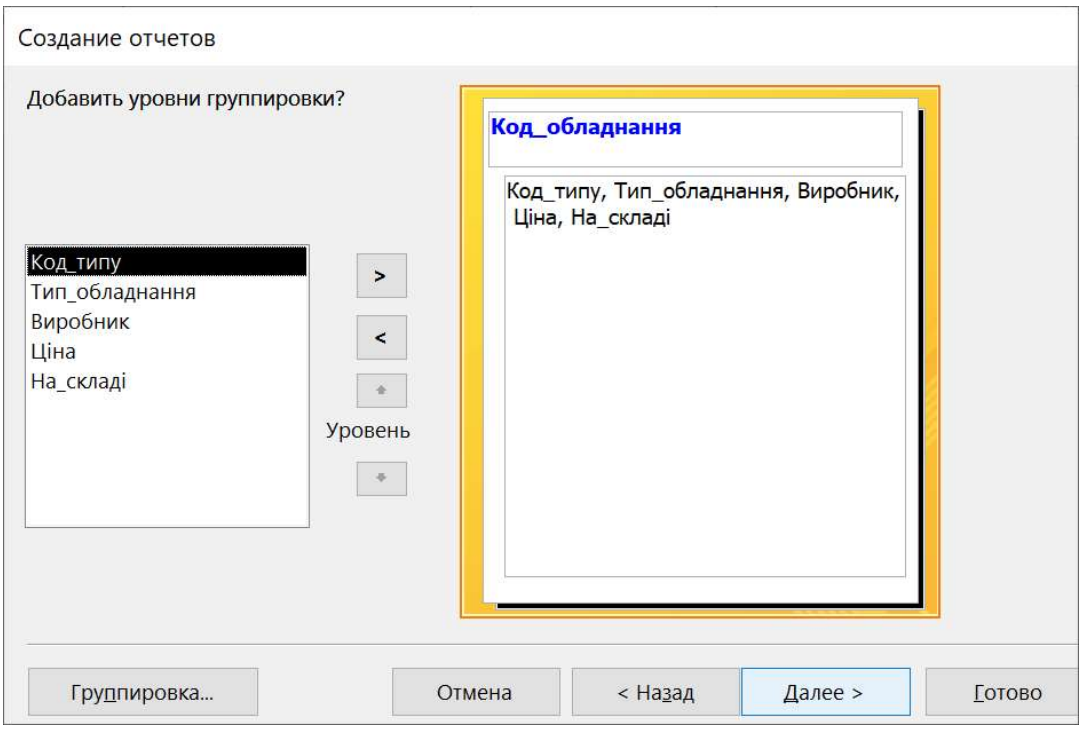



Рис. 11.6. Вибір поля для групування даних у звіті

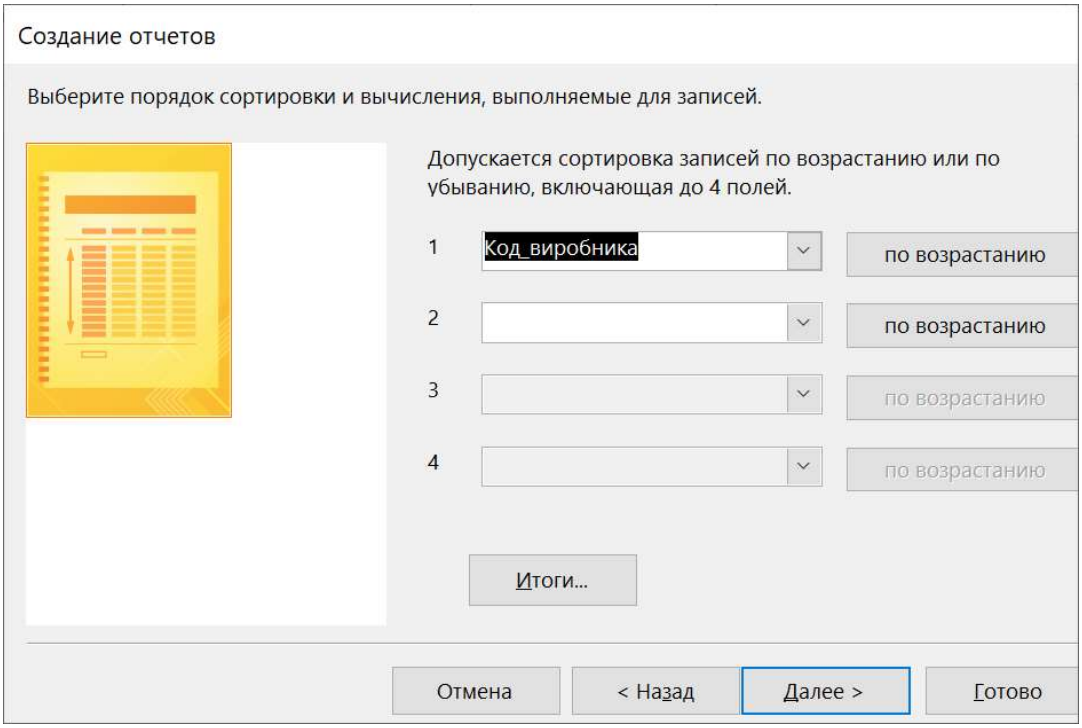
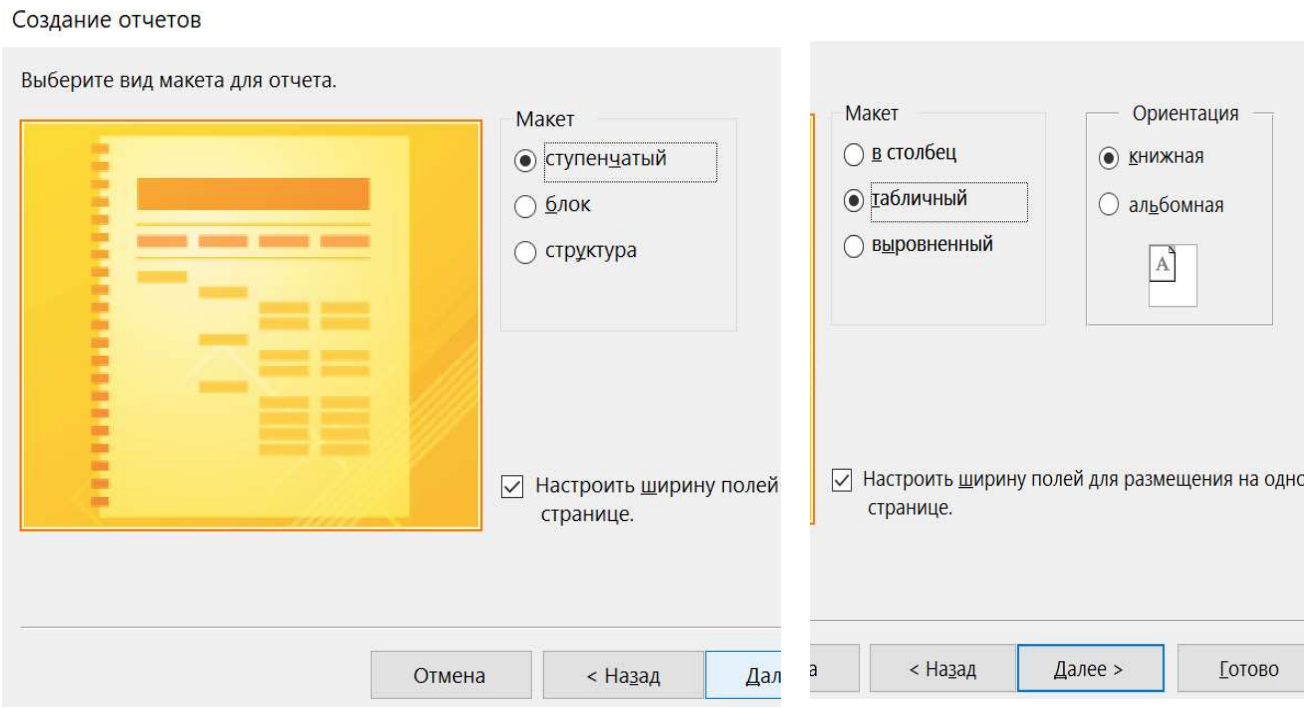



Рис. 11.7. Вибір впорядкування по заданому полю




а) б)

Рис. 11.8. Варіанти для вибору зовнішнього вигляду звіту

На останньому етапі створення звіту задається ім’я звіту та пропонується переглянути звіт (рис. 11.9), створений з вибором параметрів на попередніх кроках, або переглянути макет звіту і за потребою внести зміни до макету звіту. Наприклад, змінити розміри полів для виведення даних (рис. 11.10).

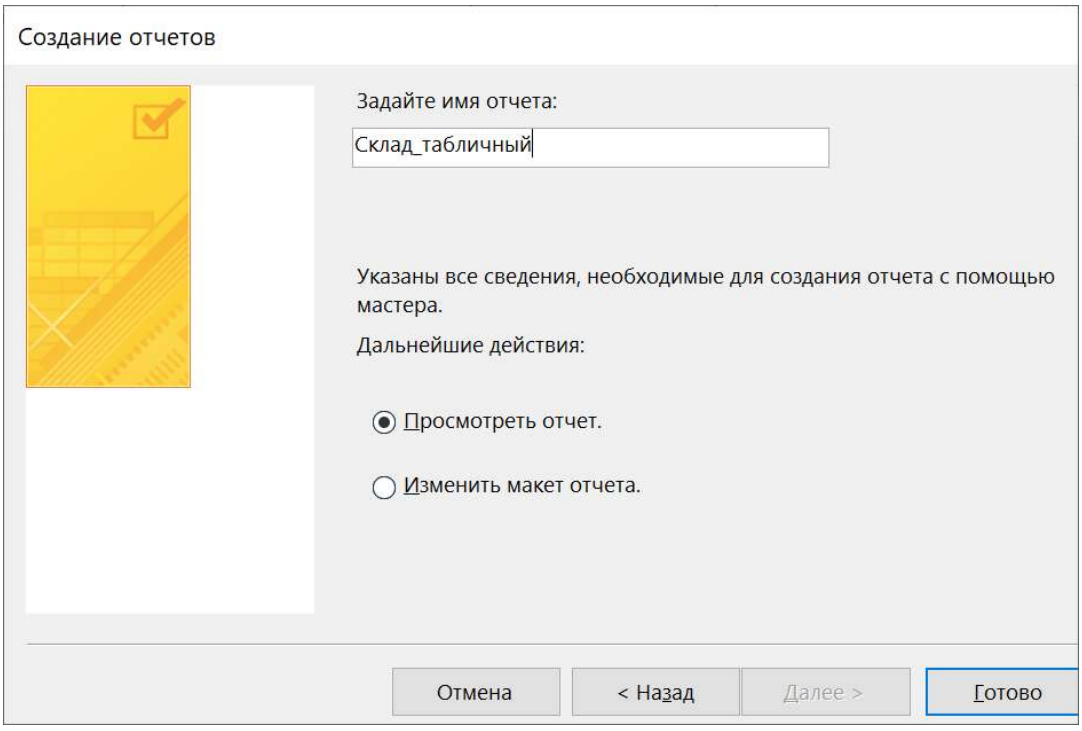
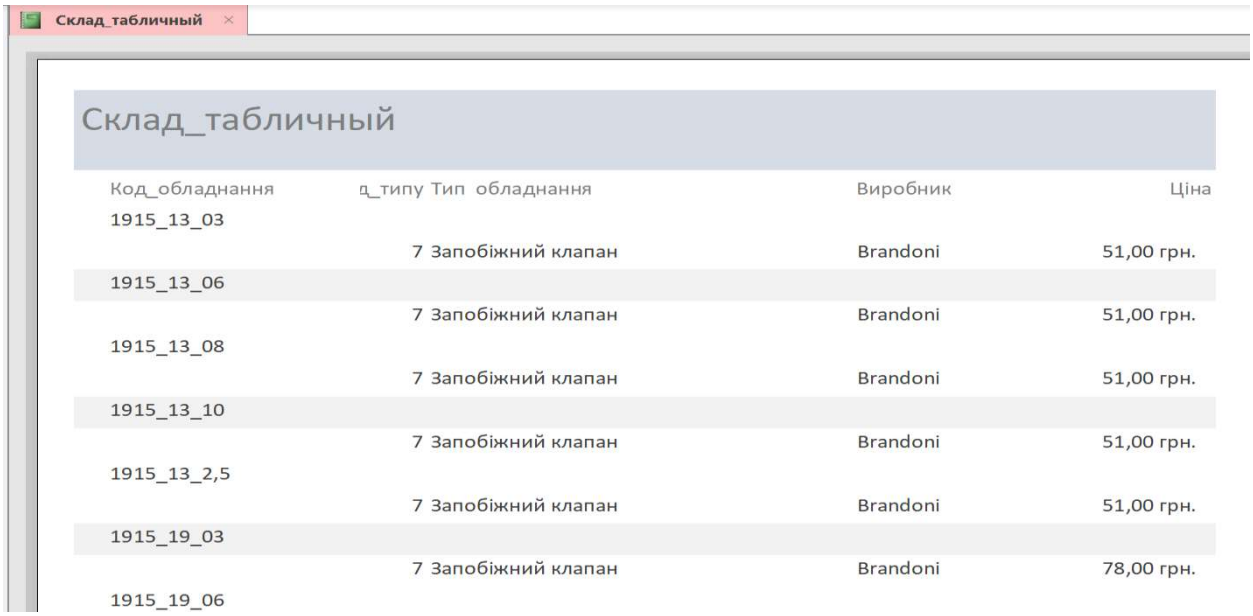



Рис. 11.9. Вибір перегляду звіту чи зміни макету звіту



Склад_табличный

| Код_обладнання | д_типу | Тип обладнання | Виробник | Ціна |
|---|---|---|---|---|
| 1915_13_03 | | | | |
| | 7 | Запобіжний клапан | Brandoni | 51,00 грн. |
| 1915_13_06 | | | | |
| | 7 | Запобіжний клапан | Brandoni | 51,00 грн. |
| 1915_13_08 | | | | |
| | 7 | Запобіжний клапан | Brandoni | 51,00 грн. |
| 1915_13_10 | | | | |
| | 7 | Запобіжний клапан | Brandoni | 51,00 грн. |
| 1915_13_2,5 | | | | |
| | 7 | Запобіжний клапан | Brandoni | 51,00 грн. |
| 1915_19_03 | | | | |
| | 7 | Запобіжний клапан | Brandoni | 78,00 грн. |
| 1915_19_06 | | | | |

Рис. 11.10. Вигляд звіту, створеного з використанням Майстра звітів для запиту Склад

**Приклад 11.3**. Створити звіт з використанням режиму **Конструктор отчетов** для таблиці Затвори_клапани, виконати групування по полю Приєднання , впорядкування по полю Діаметр. Скрін результату записати до протоколу. Створення звіту в режимі Конструктор отчетов починається з вибору опцій **Создание/Конструктор отчетов** (рис. 11.11). Після чого з’являється вікно з макетом конструктора звітів, на якому розташовані три основні розділи: верхній колонтитул, область даних та нижній колонтитул. Призначення кожного з цих розділів вивчіть самостійно.

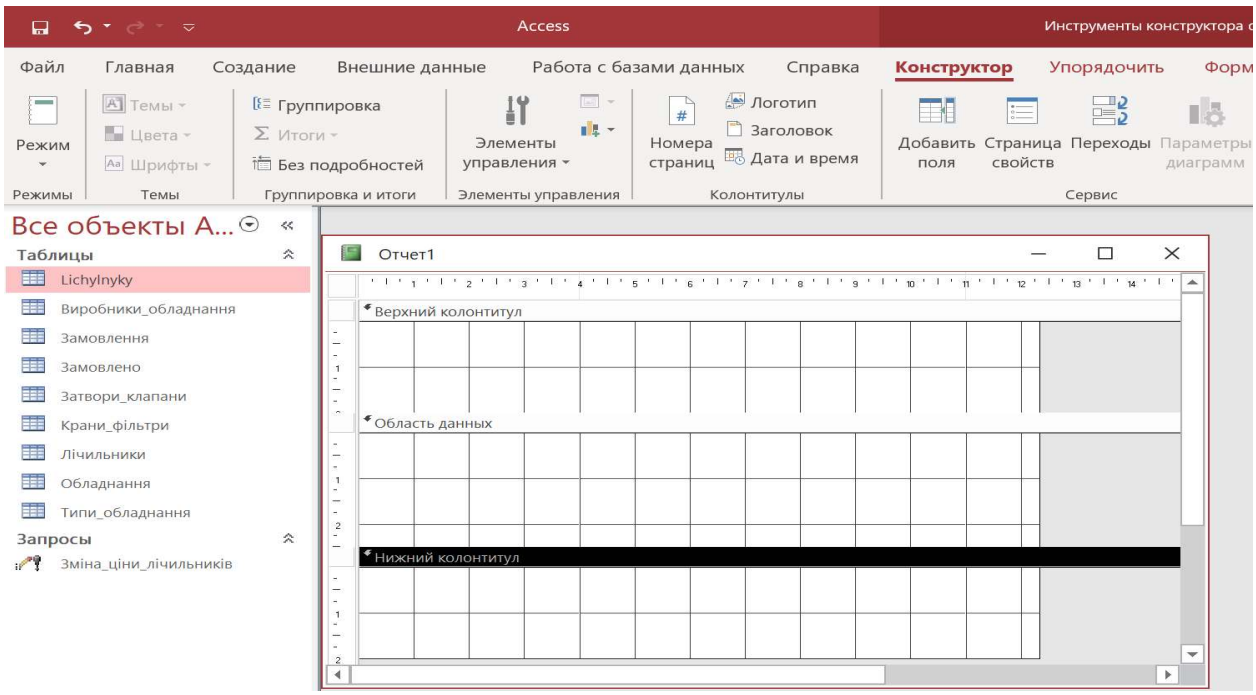



Рис. 11.11. Вигляд макета звіту в режимі Конструктор звітів

1. В верхній колонтитул вставити елемент керування **Надпись** з панелі **Элементы управления та** наберіть текст (рис. 11.12).
2. Натисніть піктограму Добавить поля. У вікні, що відкриється, оберіть потрібну таблицю та розташуйте у Області даних обрані поля (подвійний клік на імені поля і воно з'являється Області даних). Необхідно додати поля Код_обладнання, Приєднання, Діаметр. Виконайте групування по полю Приєднання( для цього обрати піктограму **Группировка)** та сортування по полю Діаметр (рис. 11.12). Після цього перетягніть поле Приєднання у область Заголовок групи Приєднання. Перегляньте результат, обравши режим Предварительній просмотр.
3. Закрити створений звіт та зберегти звіт під назвою **Затвори_клапани_звіт_конструктор**.

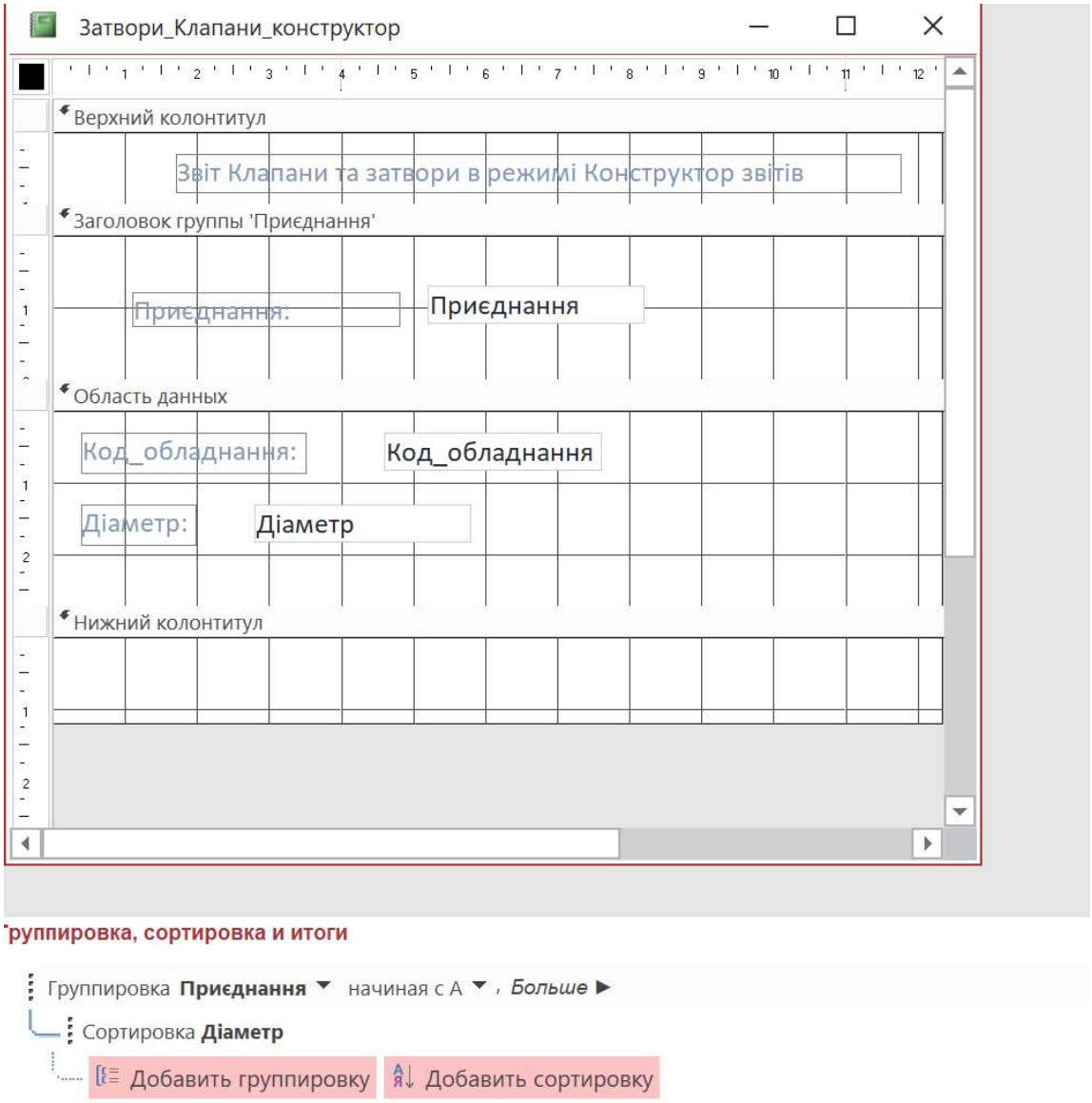



Рис. 11.12. Виконання групування та сортування по заданим полям

**Приклад 11.4.**

Створити запит на підставі таблиць Склад_обладнання, Типи обладнання, Виробники обладнання. Запит повинен містити такі поля: Код_обладнання, Код_типу, Тип_обладнання, Виробник, Ціна, Кількість_на_складі. Зберегти запит під назвою Склад_крани.

4. Створити звіт з використанням Мастер отчетов (блок) для запиту **Склад_крани**, виконати групування по полю **Виробник** , впорядкування по полю **Ціна**.
5. Створити поле **Вартість** у області даних звіту. Для цього з панелі елементів перетягуємо об’єкт Поле у область даних звіту. Перейменувавши його підпис на Вартість, записати у полі вираз

=Ціна*На_складі. Таким чином, у створеному полі буде обчислюватися вартість кожного різновиду відповідного типу обладнання, що знаходиться на складі.

6. Зберегти звіт під назвою **Склад_крани_блок**. Скрін результату записати до протоколу

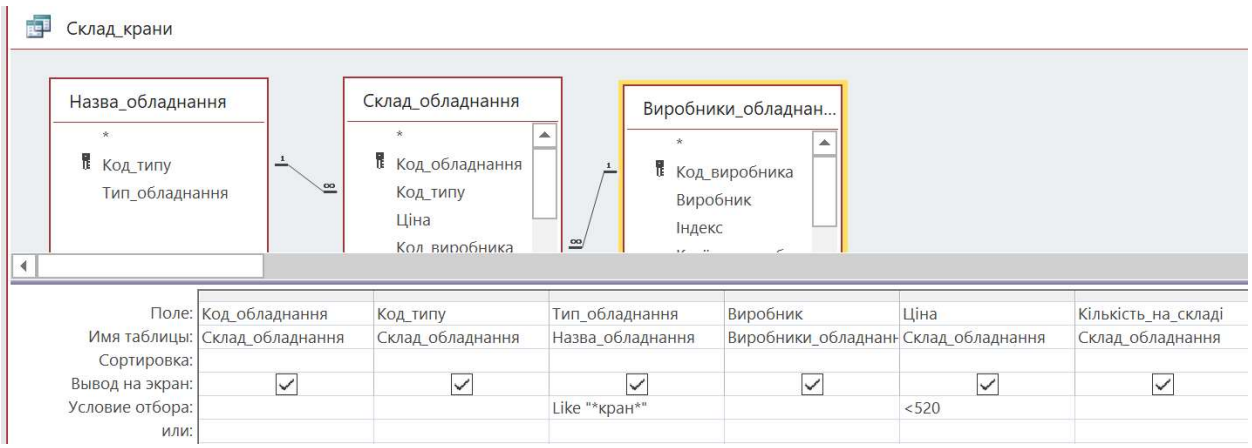



Рис. 11.13. Запит **Склад_крани** в режимі конструктора

Створити аналогічно у розділі **Примечание группы** поле та змінити його підпис на **Вартість всього**. Записати у створеному полі наступний вираз **=Sum(На_складі*Ціна),** який буде рахувати загальну вартість обладнання відповідного виробника (рис. 11.14).

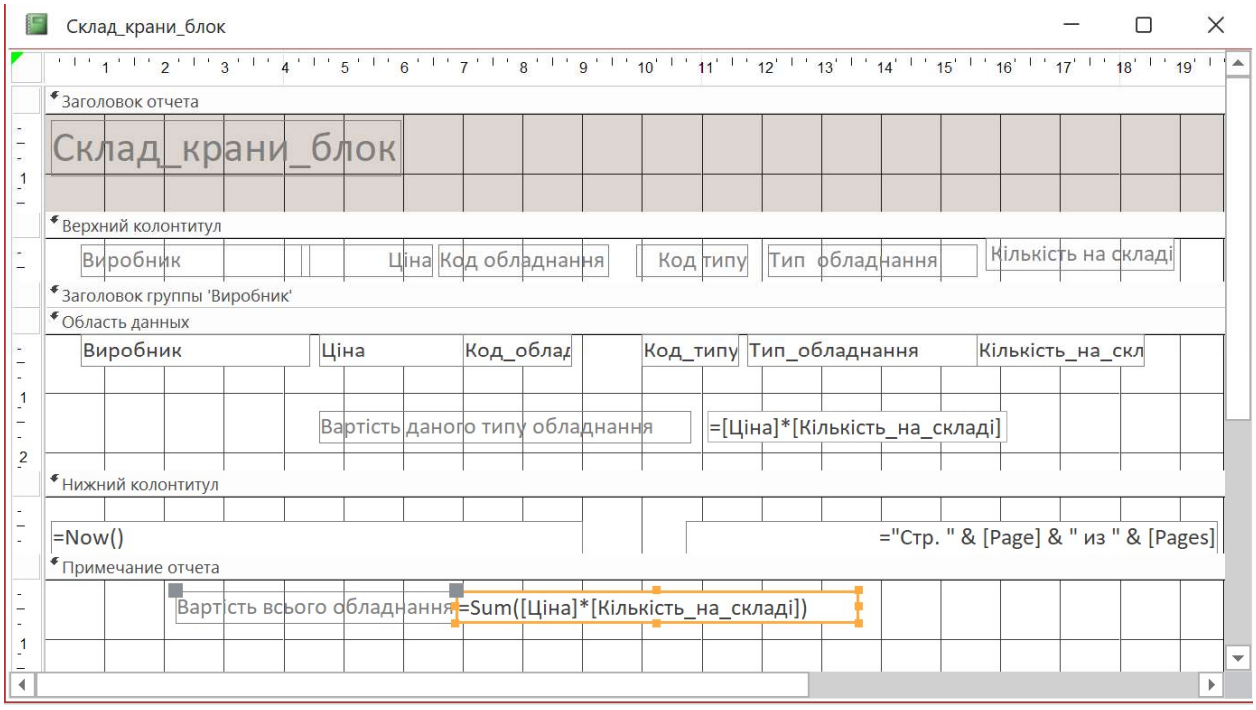



Рис. 11.14. Звіт **Склад_крани_блок** в режимі конструктора

**Приклад 11.5.** Припустимо, необхідно створити звіт про наявність лічильників газу на складі. Для інформативності краще створити звіт, який використовує підлеглий звіт. Перш за все потрібно розробити базовий запит. Перейти на закладку **Создание/ Конструктор запросов** вікна бази даних і створити новий запит, використавши таблиці **Виробники обладнання**, **Склад_обладнання** та **Назва_обладнання**.

2. Перенести поля: Код_обладнання, Код_типу, Тип_обладнання, Виробник, Ціна, Кількість_на_складі. Задати умови згідно до рис. 11.15.

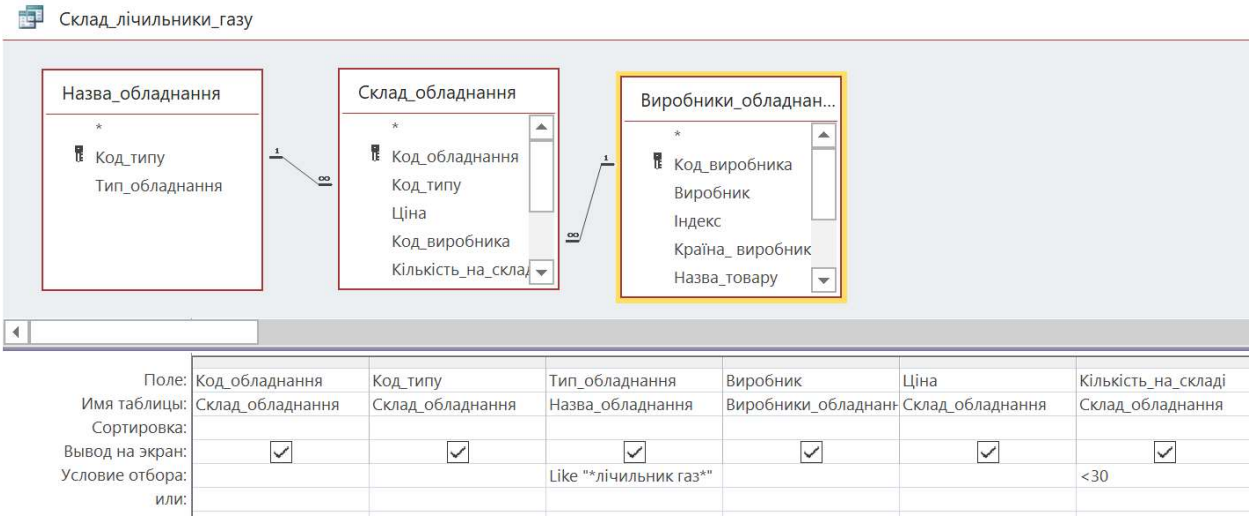



Рис. 11.15. Запит **Склад_лічильник_газу** в режимі конструктора

3. Зберегти запит під назвою **Склад_лічильник_газу**.

4. Перейти на закладку **Отчёты** та відкрити в конструкторі **Конструктор отчетов** новий звіт. В якості джерела даних звіту обрати розроблений раніше запит **Склад_лічильник_газу** (для підлеглого звіту). Прибрати **Колонтитулы** із звіту і задати відображення розділів **Заголовок отчета/Примечание отчета**. Область **Примечание отчета** в подальшому не використовується, тому потрібно за допомогою миші перетягнути його нижню межу до верхньої, встановивши таким чином для нього нульову висоту (рис. 11.16).

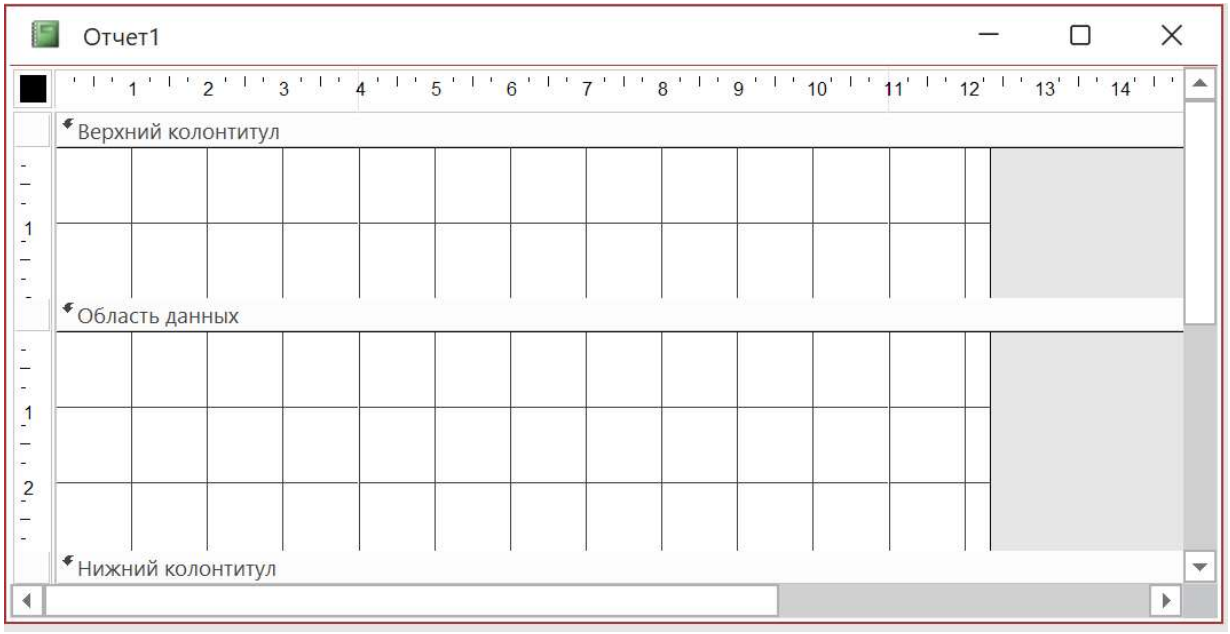



Рис. 11.16 Звіт режимі конструктора

5. Із списку полів звіту перетягнути у розділ **Область данных** наступні поля: **Код_типу**, **Тип_обладнання**, **Виробник**.

6. Створити обчислювальне поле **Вартість** (в новоствореному полі записати вираз **=[Ціна]*[Кількість_на_складі]**).

7. Послідовно вирізати всі приєднані надписи і вставити їх у розділ **Заголовок отчета** (рис. 11.17). Виділити всі поля та підписи, задаючи для них непрозору межу, використовуючи кнопку з випадаючим меню  .

8. Зберегти звіт під іменем **Склад_оперативний** та закрити.

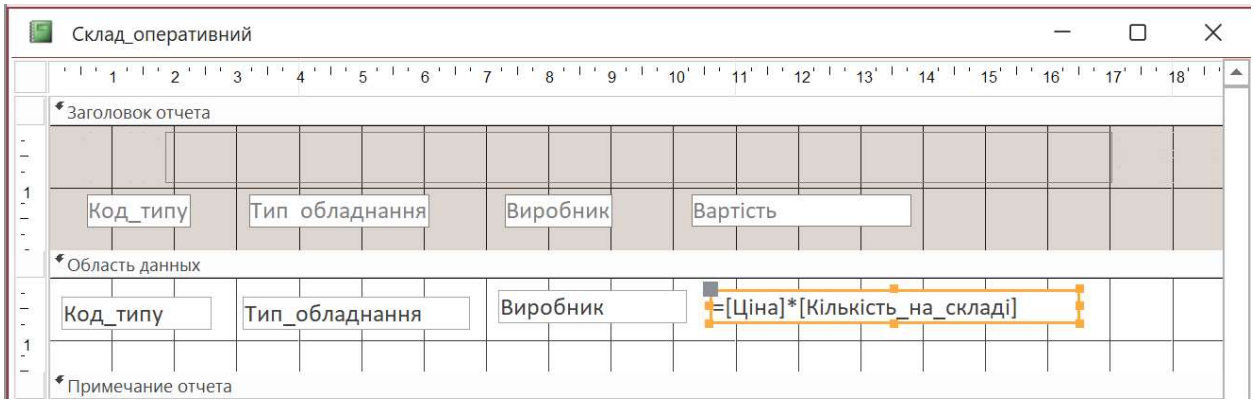



Рис. 11.17. Звіт Склад_оперативний в режимі конструктора

**Приклад 11.6.** Створити новий звіт в режимі конструктора, обравши в якості джерела даних таблицю **Виробники _обладнання**.

1. Аналогічно попередньому прикладу прибрати з макету **Колонтитулы**, задати відображення **Заголовок отчета/Примечание отчета**, для **Примечание отчета** задати нульову висоту.

2. В розділі **Заголовок отчета** розмістити поле **Надпись**, задаючи для нього текст **Замовлення**, а також **Поле**, задаючи для нього **Средний формат даты** і в якості даних – вираз: **=Date()**. Нижче цих елементів керування розмістити горизонтальну лінію (рис. 11.18).

3. Перетягнути зі списку полів у розділ **Область данных** наступні поля: **Виробник**, **Країна виробника**.

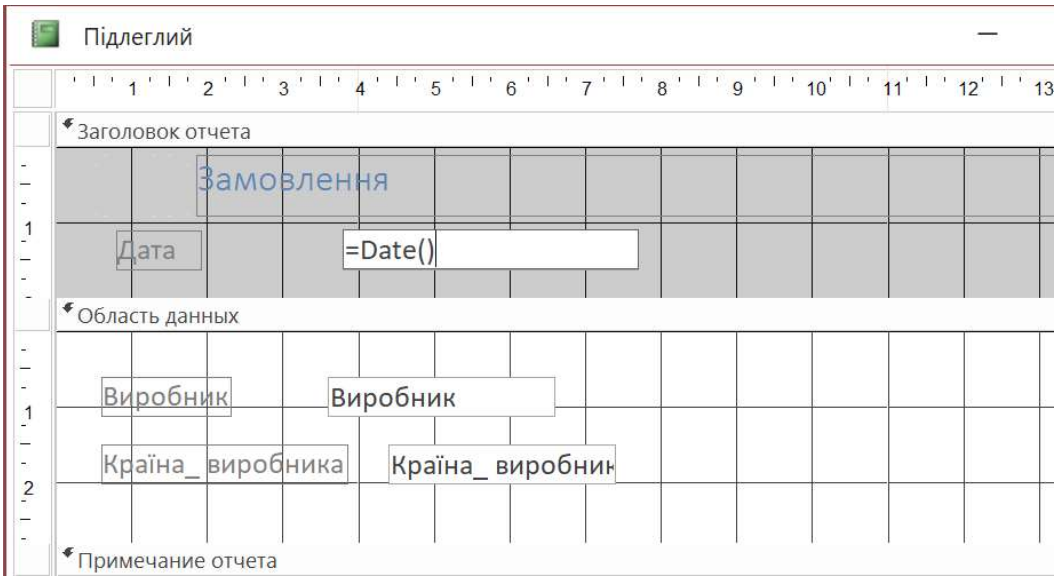



Рис. 11.18. Звіт **Підлеглий** в режимі конструктора

4. Навколо групи перерахованих полів розмістити прямокутник, щоб надати інформації остаточний вигляд. Для того щоб прямокутник не закривав дані, необхідно обрати для властивості цього прямокутника **Тип фона** значення **Прозрачный** (рис. 11.19).

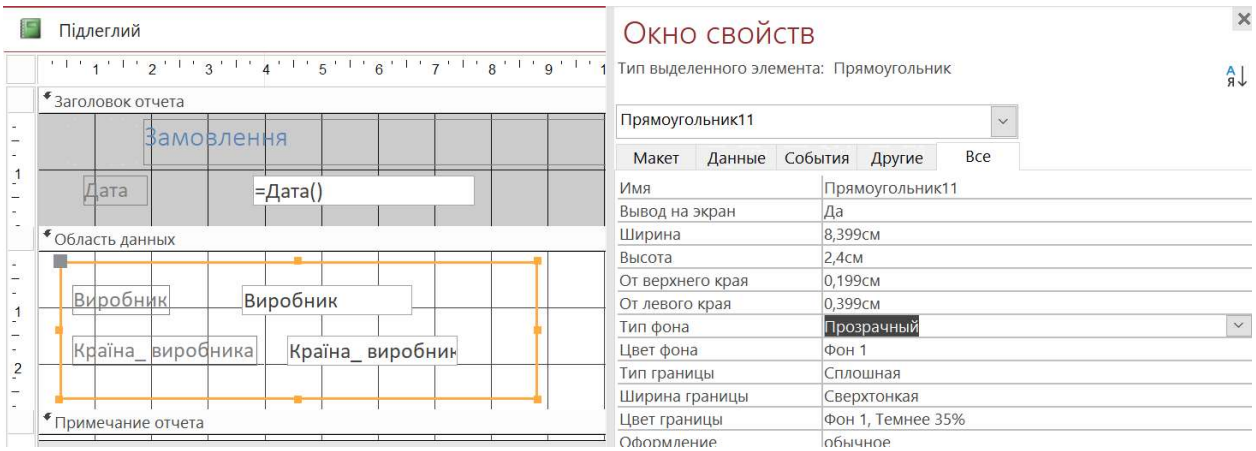



Рис. 11.19. Вибір значення **Прозрачный** для властивості прямокутника **Тип фона**

5. Вибрати на панелі елементів **Подчиненную форму/отчет** і натиснути мишею в **Область данных** нижче інформації виробника. Викликати властивості підлеглого звіту, де в пункті **Все** в рядку **Объект-источник** вказати звіт **Склад_оперативний** в якості підлеглого звіту (рис. 11.20, рис. 11.21). Результат виконання завдання у конструкторі наведений на рис. 11.22.

6. Зберегти звіт під назвою **Підлеглий**.

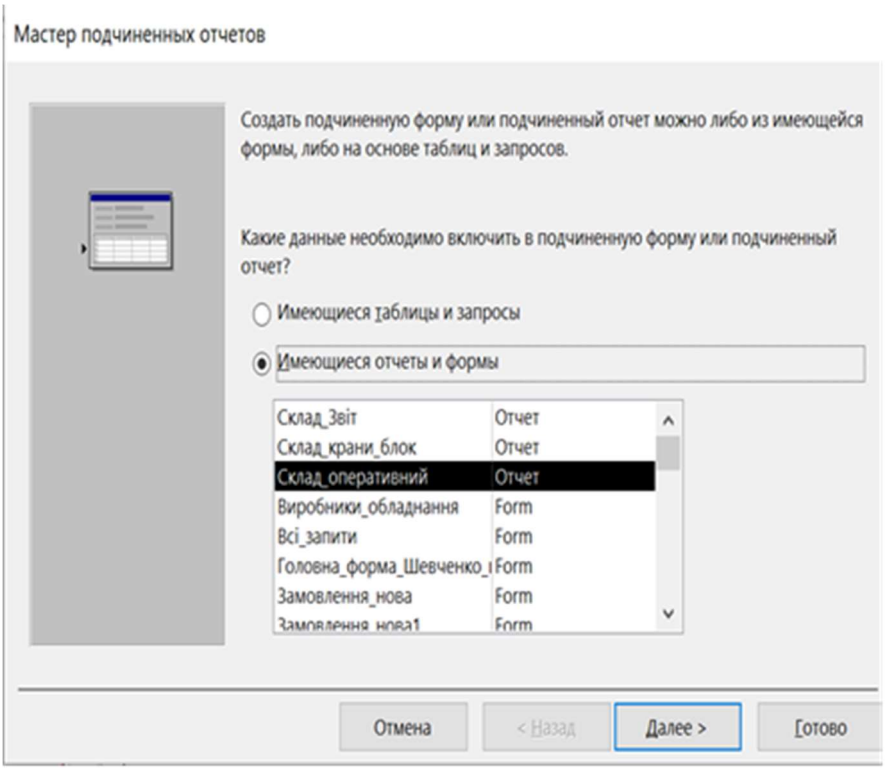



Рис. 11.20. Вибір джерела даних підлеглого звіту

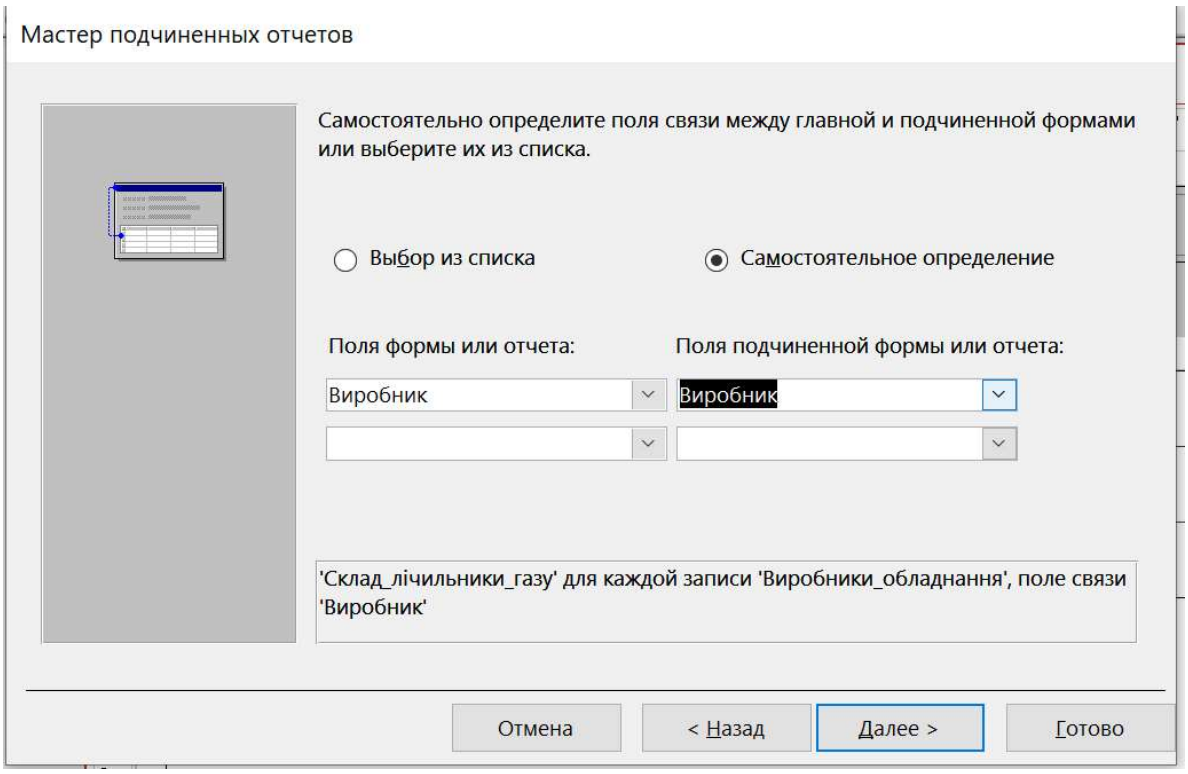



Рис. 11.21. Вибір поля для встановлення зв'язку основного звіту з підлеглим

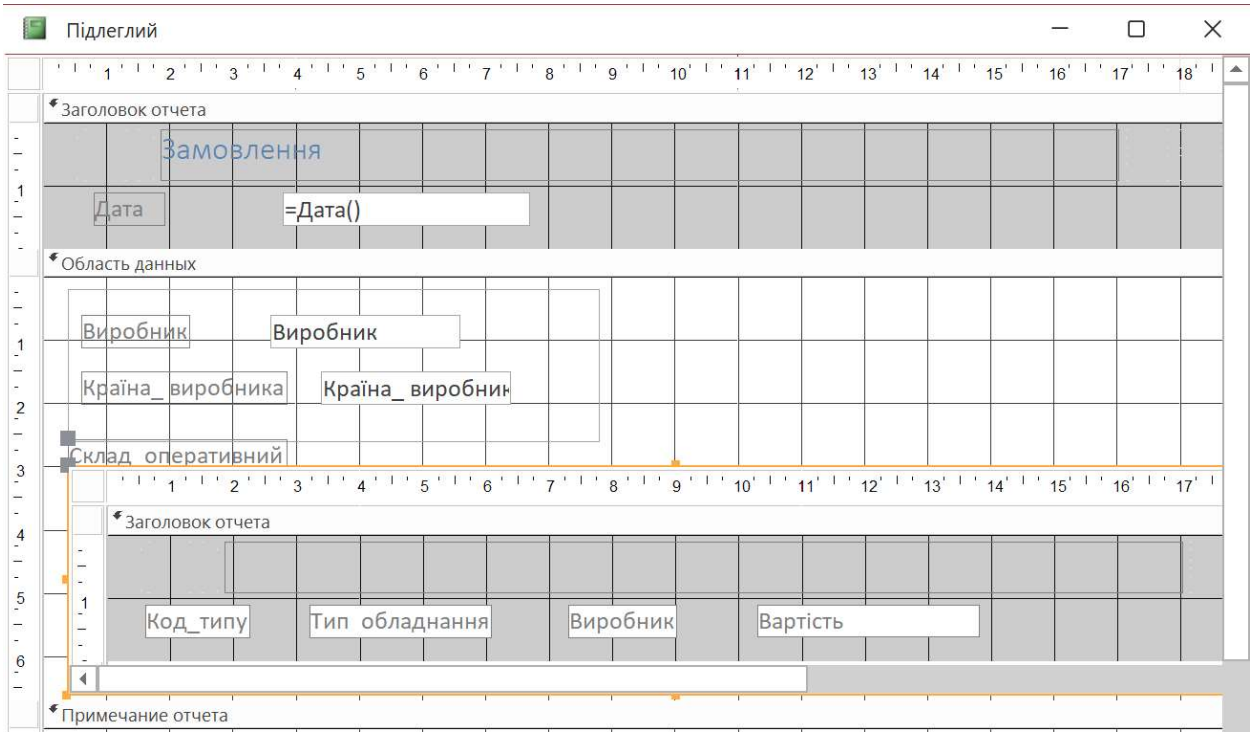



Рис. 11.22. Звіт з підлеглим звітом у режимі конструктора

**Приклад 11.7.** Створити звіт з діаграмою.

1. Модифікувати запит **Склад_лічильник_газу,** вигляд запиту в режимі конструкторат представлено на рис. 11.23 та зберегти його з ім'ям **Склад_лічильник_газу_д.**

2. На базі запиту **Склад_лічильник_газу_д** створити запит, який буде відображати наявність лічильників на складі та обчислювати їх вартість для кожного типу лічильників окремо. Для цього у режимі SQL набрати наступний текст:

SELECT [Ціна]*[Кількість_на_складі] AS Вартість, [Склад_лічильник_газу].Тип_обладнання, [Склад_лічильник_газу. Кількість_на_складі

FROM [Склад_лічильник_газу];

Зберегти запит під назвою **Запит_діаграма**.

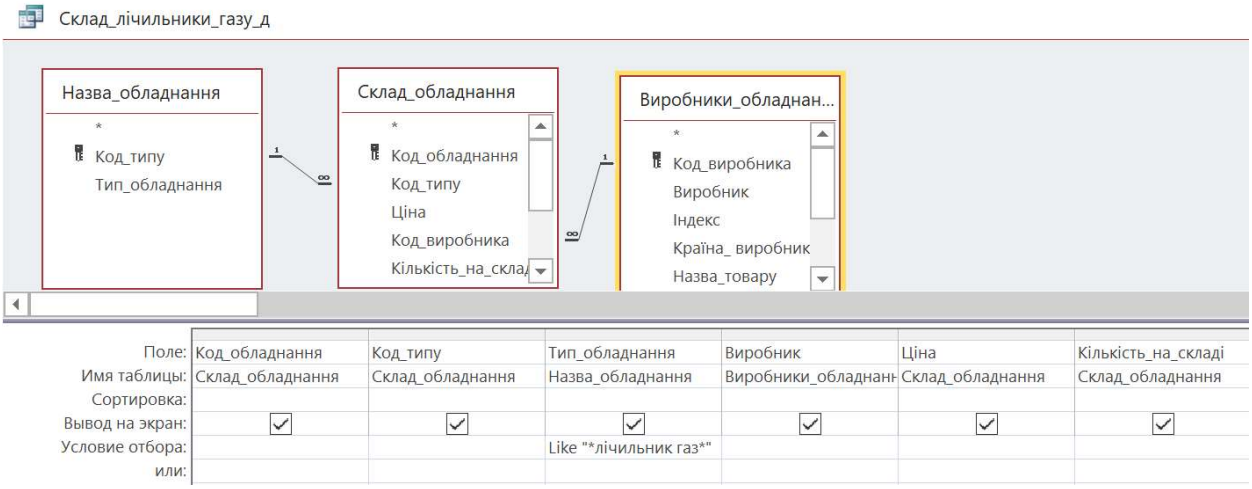
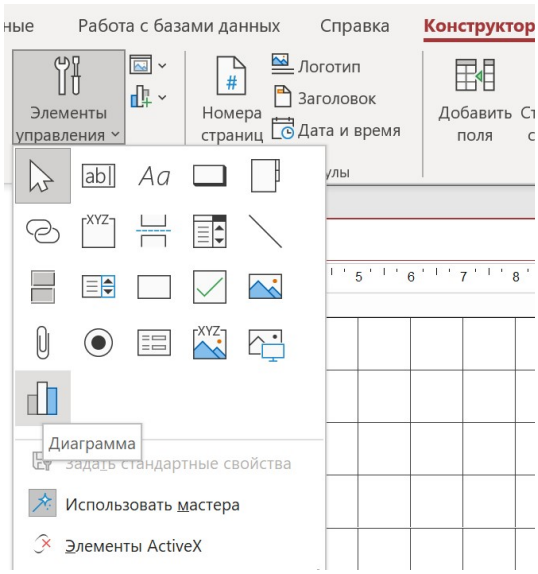


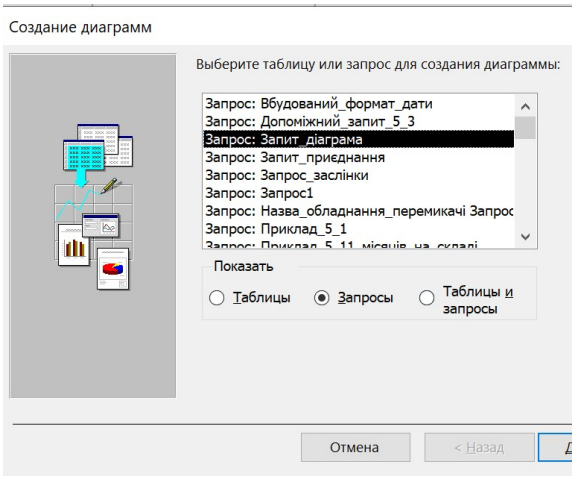

Рис. 11.23. Модифікований запит Склад_лічильник_газу у режимі конструктора

3. Перейти на закладку **Отчеты** та відкрити новий звіт в режимі конструктора. Об’єкт для створення звіту не вказувати. Видалити з його макету колонтитули та вибрати на панелі елементів керування піктограму **Диаграмма**, натиснути ліву кнопку миші в розділі **Область данных** та намалювати контури вікна, в якому буде розміщено діаграму (рис. 11.24 а).
4. Після цього буде запущений **Мастер создания диаграмм**, де в якості джерела даних вказати раніше створений запит (рис. 11.24,б).
5. Після натискання кнопки **Далее** на екрані з’являється вікно, зображене на рис. 11.24 б. Поля з закладки **Доступные поля** перетягнути у закладку **Поля диаграммы** (рис. 11.24,в).
6. Вибрати тип діаграми, в результаті з’являється вікно, зображене на рис. 11.24,г.

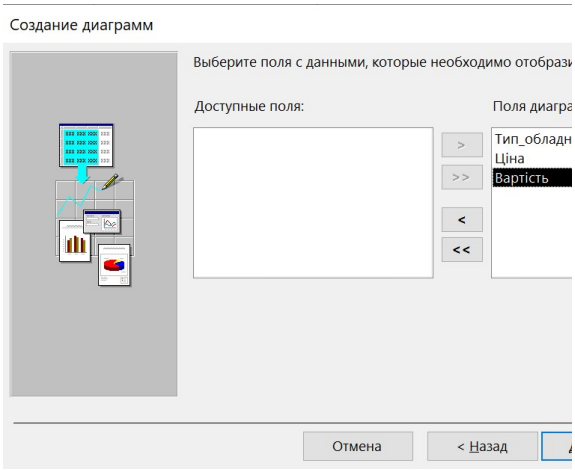
7. Задати назву діаграми. Зберегти звіт під назвою **Звіт_з_діаграмою**. Результат виконання роботи наведений на рис. 11.24,д.



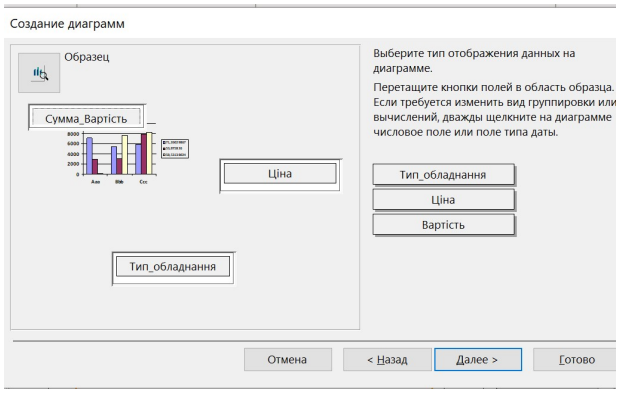

а)




б)




в)




г)


д)

Рис. 11.24

## 11.2. Завдання до комп'ютерного практикуму

1. Виконати приклади 11.1-11.7, наведені в теоретичних відомостях. Створені в прикладах звіти запам’ятати з іменами, що вказані в прикладах, додавши до кожного імені прізвище студента. Завдання прикладів, скріншоти всіх створених запитів, звітів у режимах конструктора та результати їх виконання занести до протоколу.
2. Створити звіт з використанням Мастера отчетов для таблиці Лічильники_газу. Для звіту обрати поля Умовний_прохід_мм, Ціна, Максимальна_витрата, Мінімальна_витрата, Приєднання. Групування задати по полю Приєднання, впорядкування по полю Умовний_прохід_мм. Зберегти звіт під назвою **Майстер_ім’я_таблиці_прізвище_студента.**
3. Створити звіт (блок) для запиту Затвори_запит (створити попередньо). Зберегти звіт під назвою **Блок_ім’я_джерела_прізвище_студента**.
4. Створити звіт з підлеглим для виробів: лічильники для варіантів (4,9,13,16), клапани (для варіантів 1,8,11,17), крани (для варіантів 6,10,14,18), заслінки (для варіантів 3,12,15,19), фільтри (для варіантів 2,5,7,20).
5. Створити звіт з діаграмою для відношення клапани-ціна (для варіантів 1,8,11,17), фільтри-ціна (для варіантів 2,5,7,20), заслінки-ціна (для варіантів 3,12,15,19), лічильники-ціна для варіантів (4,9,13,16), крани-ціна (для варіантів 6,10,14,18).
6. Детально описати послідовність виконання пунктів 2-5 завдання до комп'ютерного практикуму у протоколі комп'ютерного практикуму.
7. Дати повні відповіді на контрольні запитання у протоколі комп'ютерного практикуму.

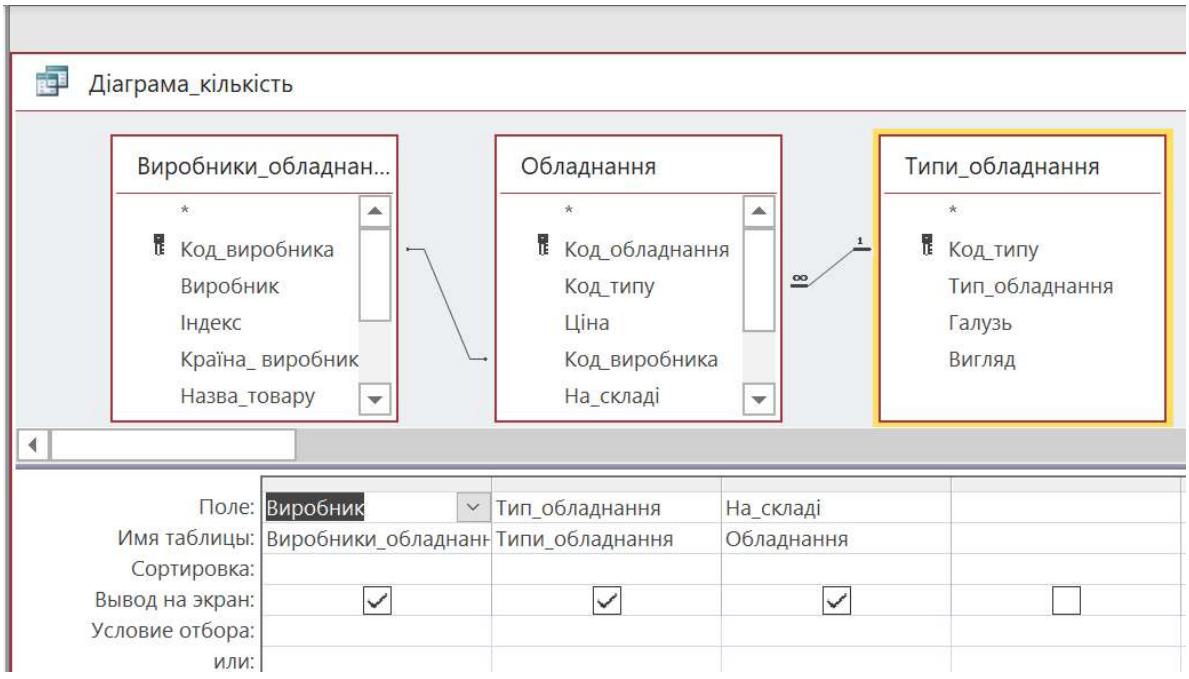
## 11.3. Контрольні запитання

1. Поясніть призначення об’єкту Access “Звіт”.
2. З яких розділів складається звіт у СУБД Access, які з них є головними, а які допоміжними?

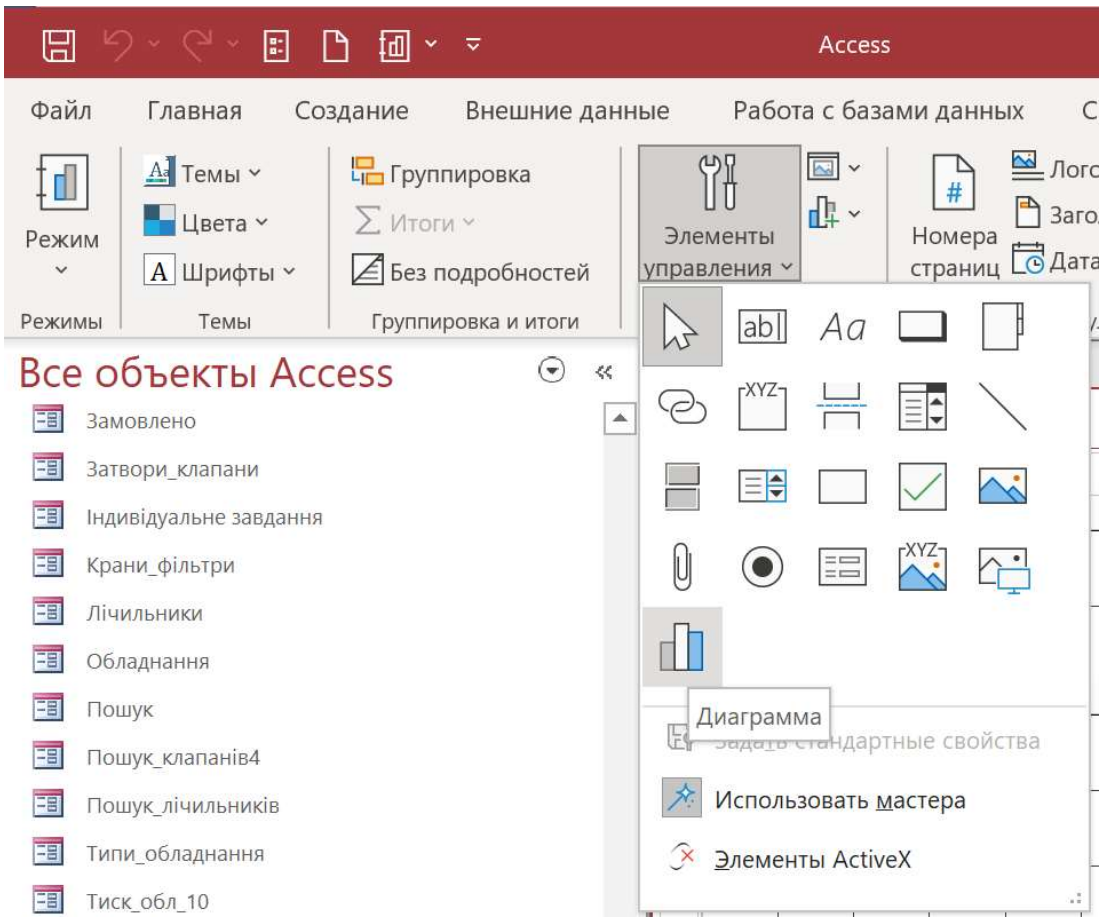
3. Наведіть властивості звітів та призначення кожного з розділів звіту.
4. Назвіть відмінності між звітами та формами.
5. Як переглянути в конструкторі звітів список полів та вставити потрібне поле в звіт?
6. Чи можливо виконати в звіті групування записів, не відображаючи “Заголовка” і (або) “Примечания группы”?
7. Де доцільно розміщувати значення поля, по якому необхідно виконати групування?
8. Де доцільно розміщувати підсумок даних по заданій групі?
9. Яка функція дозволяє включити в звіт дату?
10. Як створити у звіті поле, значення якого буде обчислюватися?
11. Яка команда дозволяє вибрати розмір сторінки звіту, її поля?
12. Як встановити потрібний інтервал між рядками з даними звіту?

## Додаток

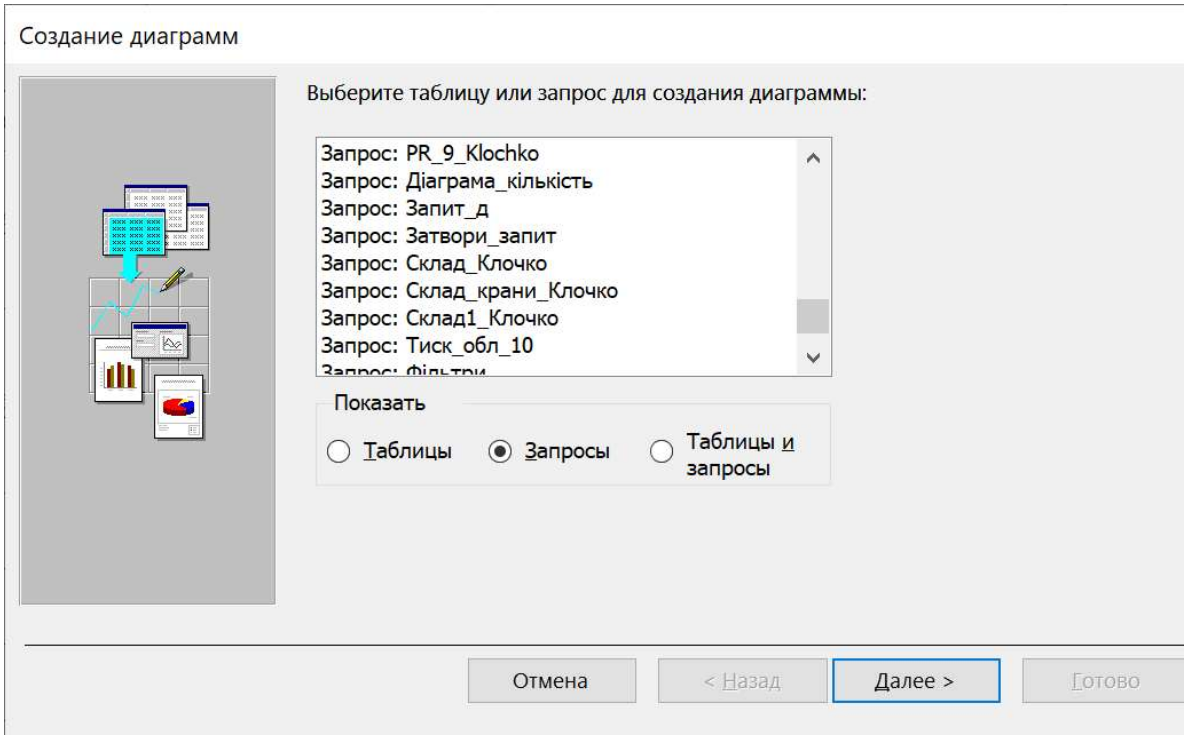
Створити запит



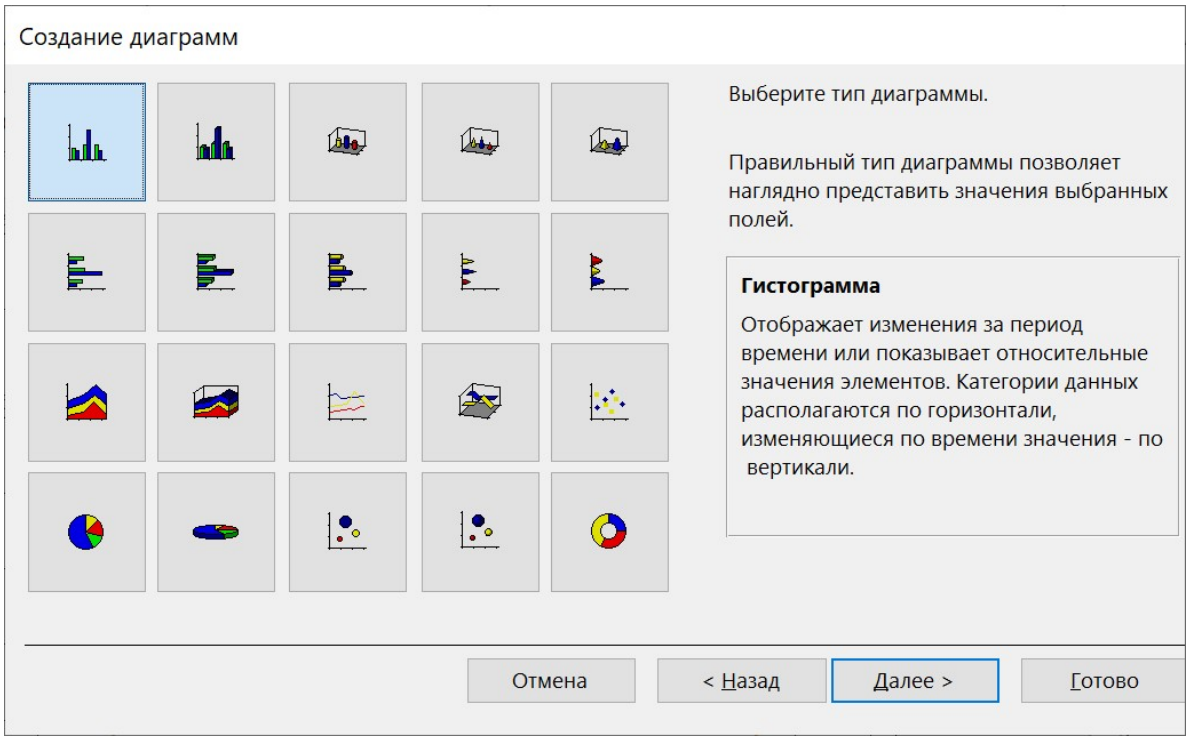

Використати запит для побудови діаграми



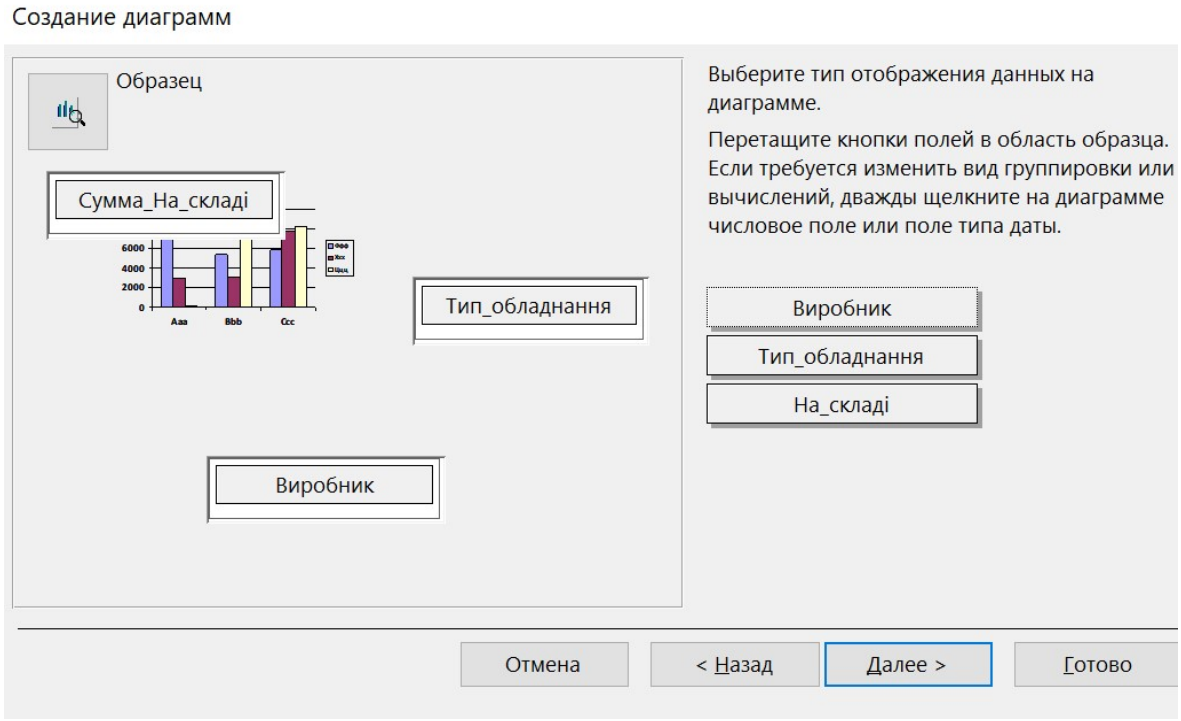

Обрати запит Діаграма_кількість



Обрати вигляд діаграми




Отримуємо зразок гістограми



Вигляд звіту

# Комп'ютерний практикум № 12. Створення опцій користувальницьких меню

*Мета роботи* - оволодіти можливостями Access по створенню головного та контекстного меню для учбової бази даних

**Теоретичні відомості**

У версіях Access, починаючи з Access 2010 для створення меню користувача використовуються макроси. В об’єктах Access може бути використано два види меню: головне (настроюване) та контекстне (локальне). Головне меню розташовується в рядку головного меню Access на вкладці Надстройки. Контекстне меню викликається на виконання правою кнопкою миші. Але для виконання опцій обох видів меню потрібно спочатку створити макроси з відповідними макрокомандами.

## 12.1. Створення настроюваного меню з використанням макросів

Створення настроюваного меню з використанням макросів розглянемо на прикладі. **Приклад 12.1.** Нехай потрібно виконати відкриття об’єктів: форми **Всі_макроси** і таблиці **Лічильники**. Реалізувати цю дію можна різними способами. Ми вокористаємо опції настроюваного меню. Назвемо ці опції меню **Відкрити_форму_Всі_макроси** та **Відкрити_таблицю_Лічильники**. Для створення опцій меню будемо створювати макрос, тому на вкладці **Создание** в групі **Макросы** необхідно натиснути кнопку **Макрос**. Після чого створимо макрос **Меню_користувача_настроюване** з двома вкладеними макросами (рис. 12.1). Кількість вкладених макросів може бути різною. На рис. 12.1 показано приклад об'єкта **Макрос** з двома в субпідрядними макросами для призначеного користувачу (головного) меню або контекстного меню.

**Порада:** Щоб створити гарячу клавішу доступу для вибору команди меню за допомогою клавіатури, необхідно ввести перед літерою назви меню, яка повинна бути клавішею доступу знак & (наприклад, &**Відкрити_таблицю_Лічильники**. Літера **В** буде підкреслюватися в меню.

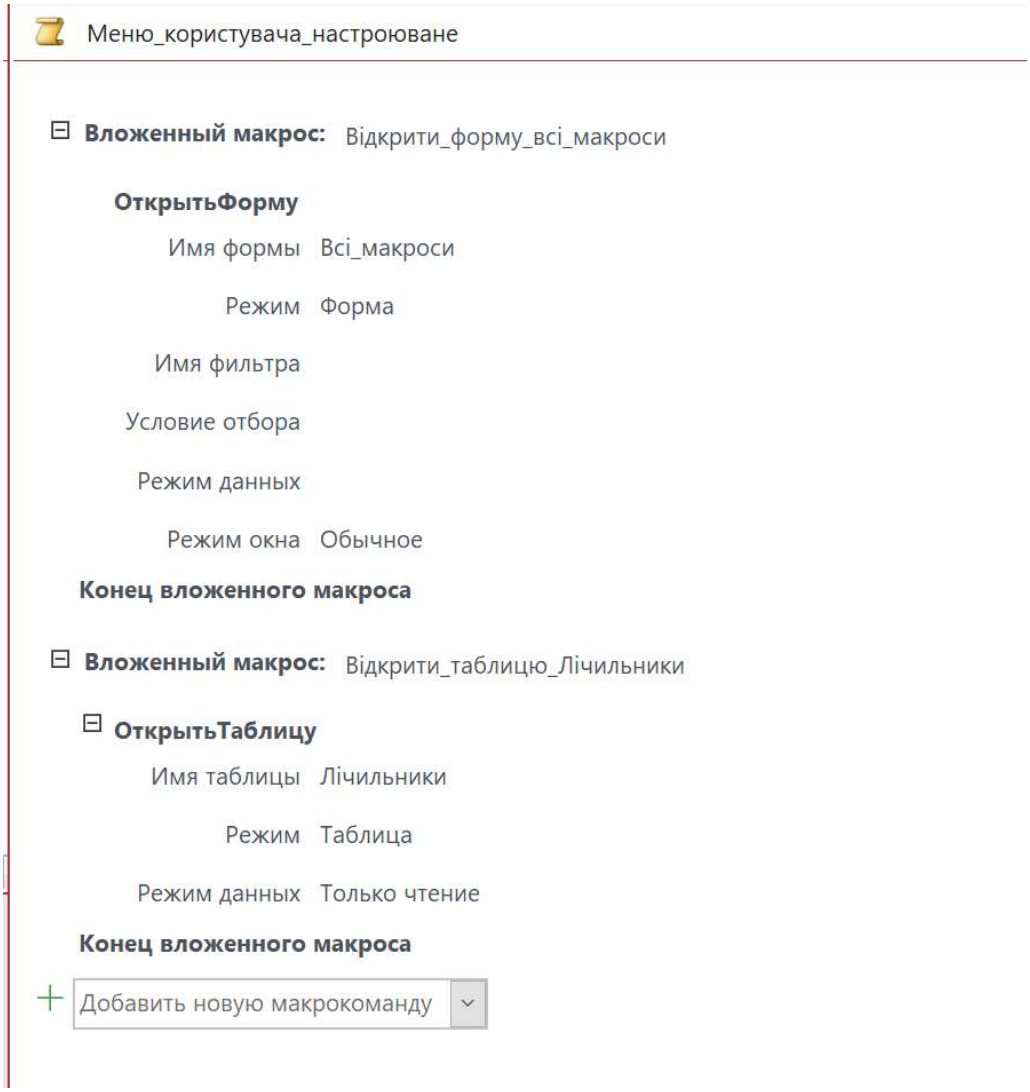



Рис. 12.1. Макрос для реалізації двох опцій меню

1. Тепер необхідно створити макрос що містить макрокоманду **Добавить меню**. У вікні конструктора макросів в полі **Додати нову макрокоманду** необхідно обрати **Добавить меню**. В поле аргументу **Имя меню** макросу **Добавить меню** введіть назву меню (наприклад, **Команди форми**). Цей аргумент не є обов'язковим, але рекомендується його задати , якщо планується додати меню на вкладку стрічки (наприклад, на вкладку **Настройки** форми або звіту). В поле аргументу **Имя макроса** необхідно ввести ім'я об'єкта **Макрос**, створеного на попередньому кроці - **Меню_користувача_настроюване**. Збережемо другий об'єкт макросу та надамо йому ім'я **Меню_стрічка**. На рис. 12.2 показаний приклад макросу меню, який створює меню, розроблене у першому пункті. Необхідно надати назву меню, в наведеному прикладі це **Тестове_меню**.

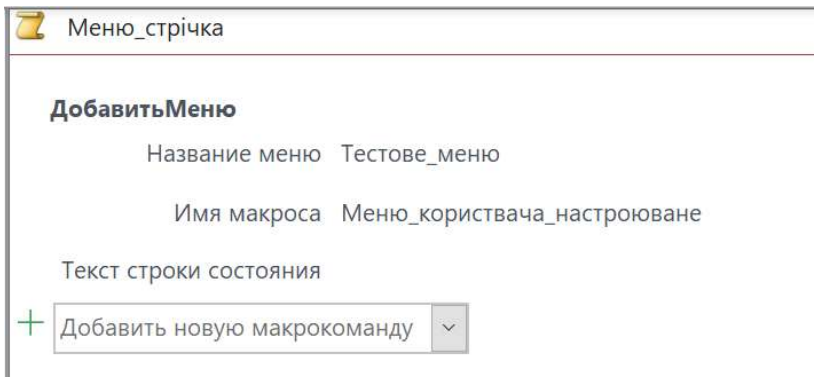



Рис. 12.2. Створення макроса меню для відповідної форми

2. Виконання створених макросів можна реалізувати з використанням головного меню або/та контекстного меню.
3. Створення головного меню, яке з'являтиметься при відкритті форми або звіту складається з наступних кроків:
    - Відкрити відповідну форму у режимі конструктора
    - У вікні **Окно свойств** на закладці **Другие** в полі **Строка меню** ввести ім'я макросу, створеного у пункті 2. В нашому прикладі це макрос **Меню_стрічка** (рис. 12.3).
    - Зберегти зміни у формі.

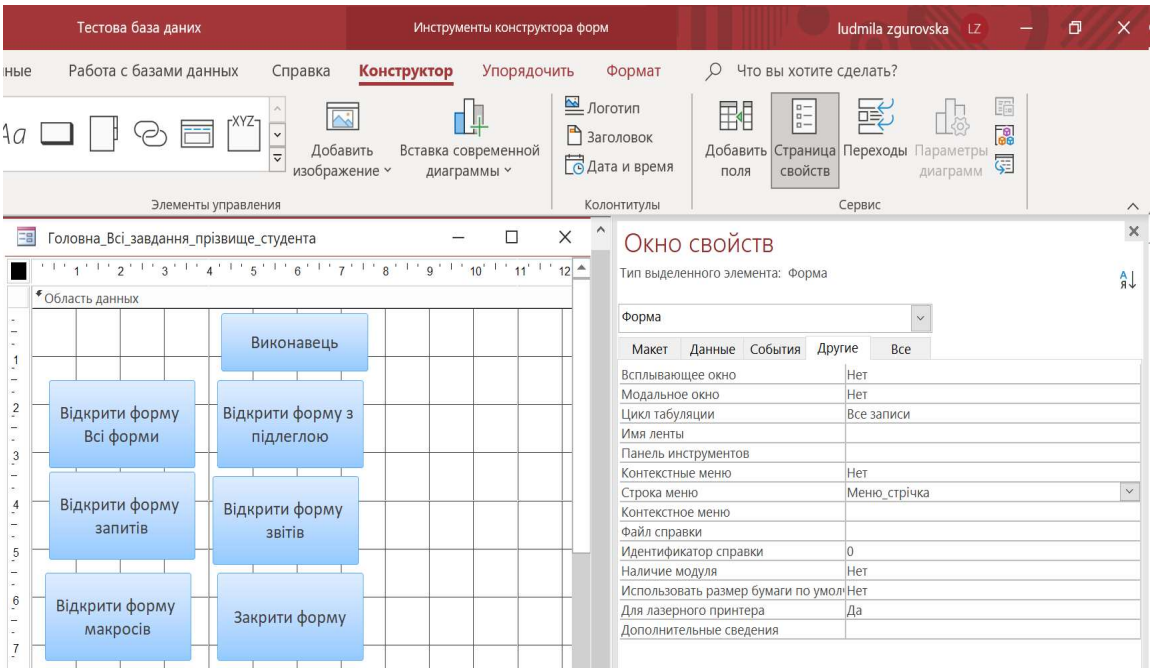



Рис. 12.3. Зв'язування макроса меню з відповідною формою

Після запуска форми на виконання в рядку головного меню Access з'являється ще один пункт **Надстройки**.

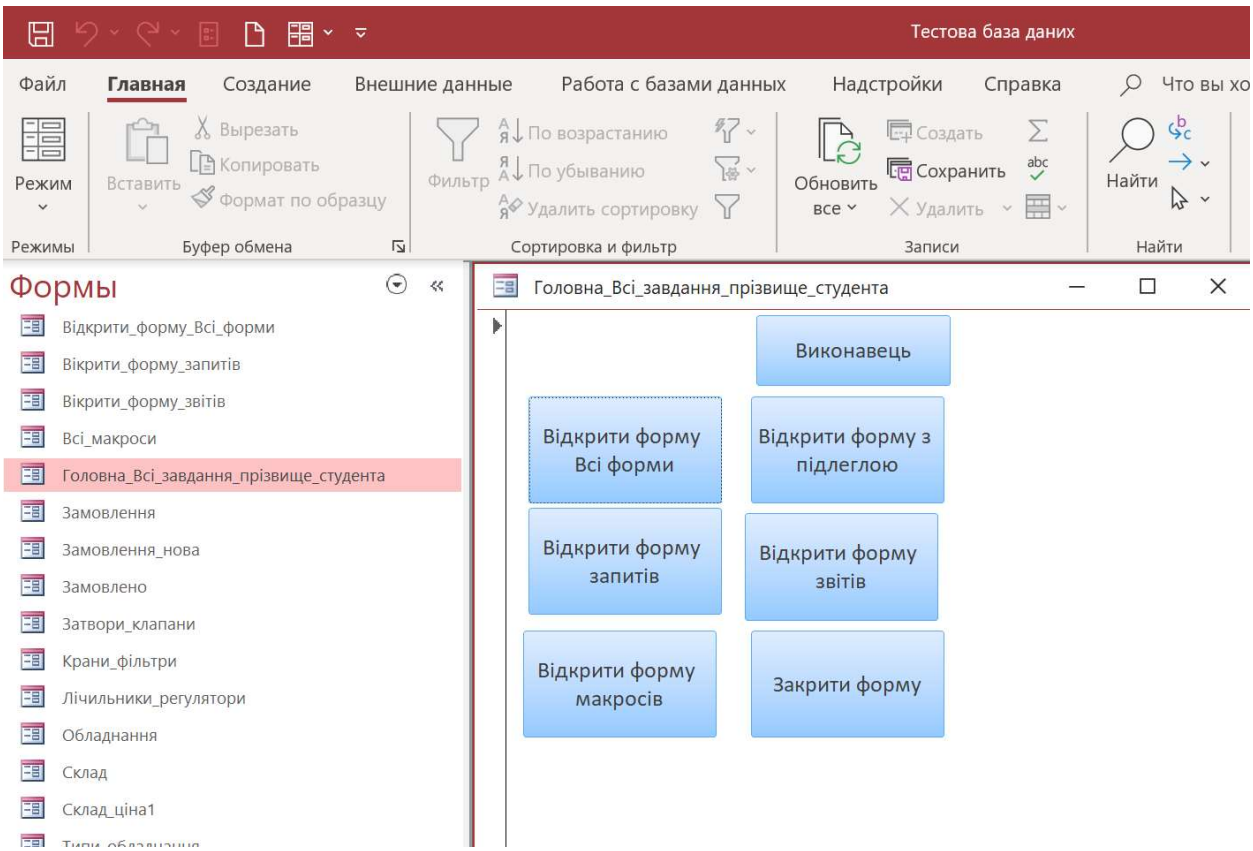



Рис. 12.4. Опція головного меню **Надстройка**

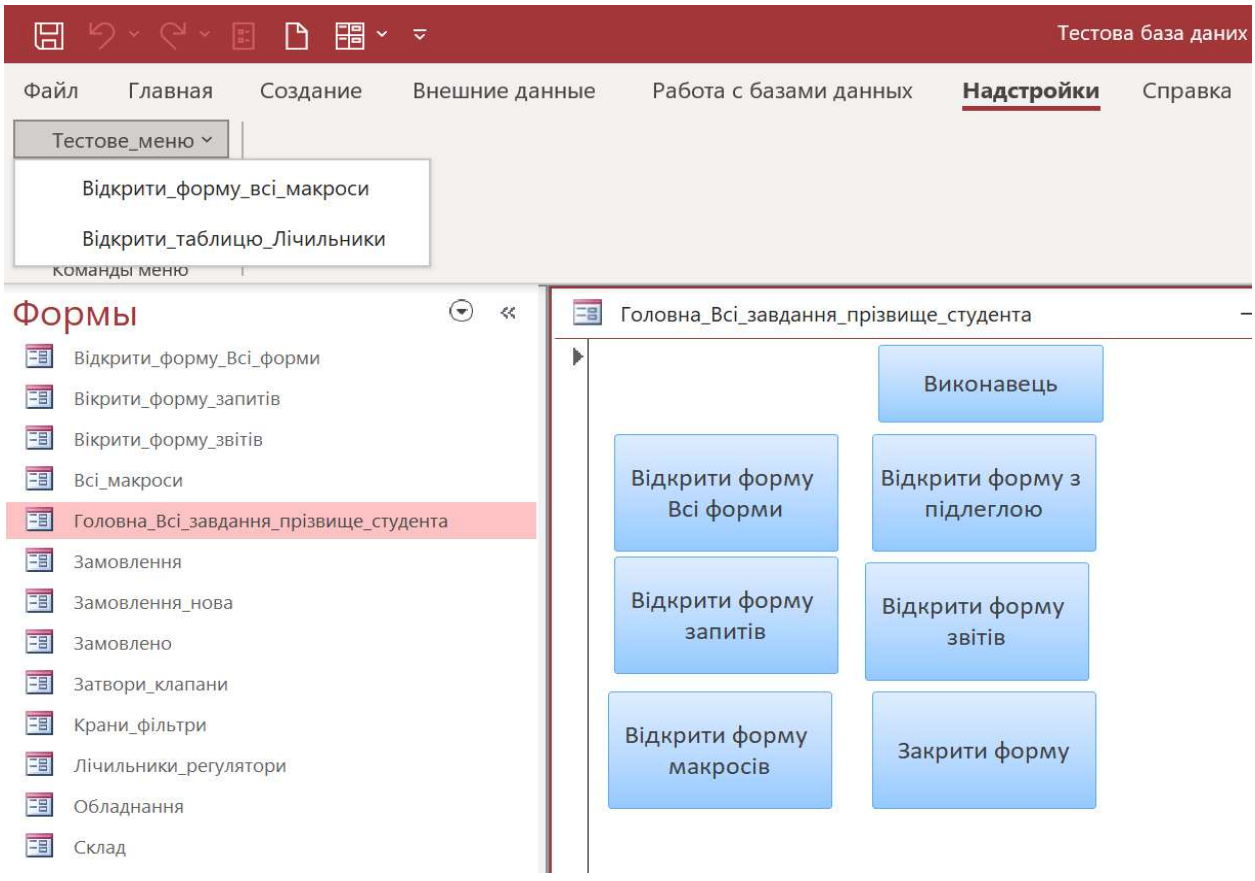



Рис. 12.5. Опції меню користувача **Тестове_меню**

При виборі опції Надстройки відкривається створене в попередніх пунктах меню Тестове_меню

## 12.2. Створення контекстного меню

1. Для реалізації прикладу використання контекстного меню створимо макрос , що запускає на виконання запити.

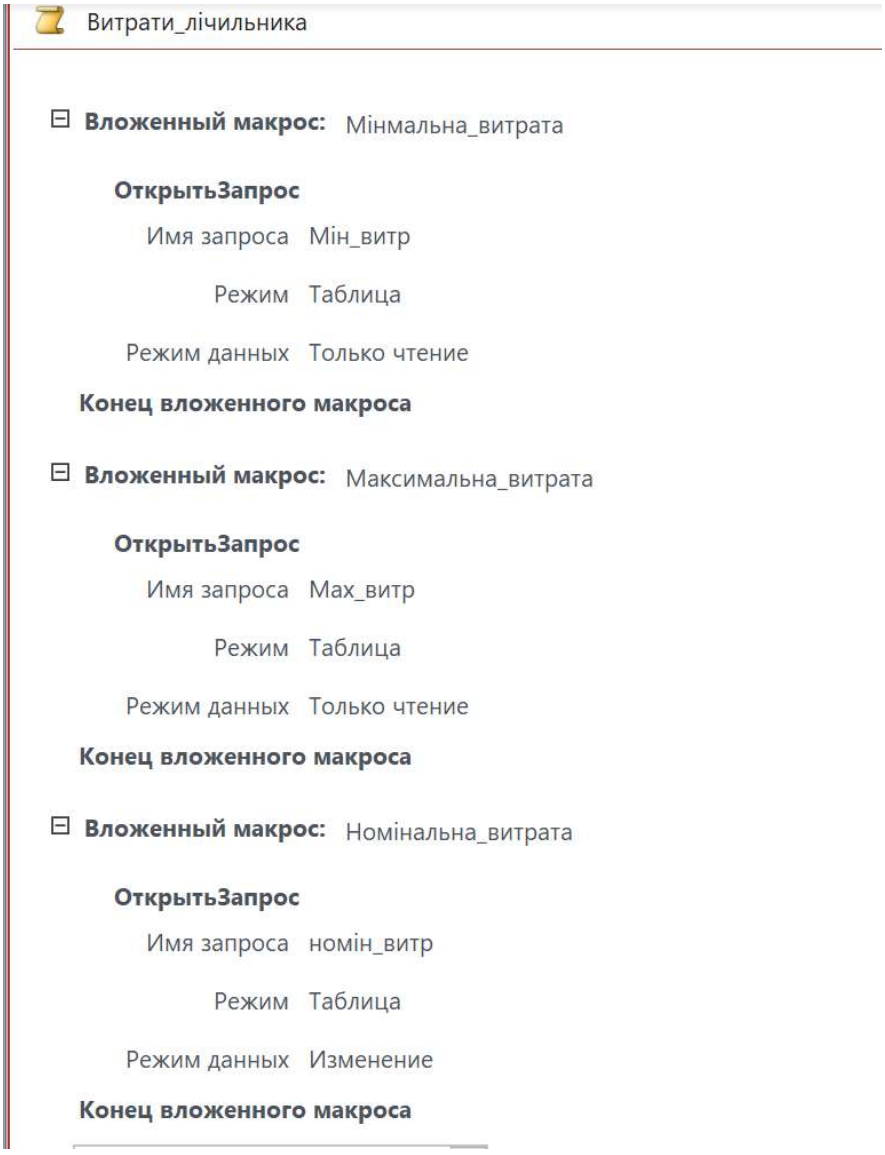



Рис. 12.6. Макрос для реалізації опцій контекстного меню

2. Тепер необхідно створити макрос що містить макрокоманду **Добавить меню**.

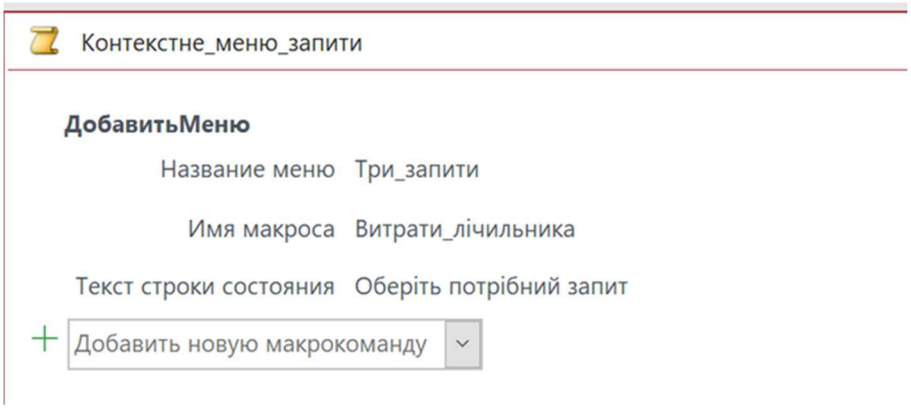



Рис. 12.7. Створення макроса меню для відповідної форми

3. Виконання створених макросів можна реалізувати з використанням контекстного меню.
4. Створення контекстного меню, яке з'являтиметься у відкритій формі або звіті при натисканні правої кнопки миші складається з наступних кроків:

- Відкрити відповідну форму у режимі конструктора.
- У вікні Окно свойств на закладці Другие в полі Контекстное меню ввести імя макросу, створеного у пункті 2. В нашому прикладі це макрос Контекстне_меню_запити (рис. 12.7).
- Зберегти зміни у формі

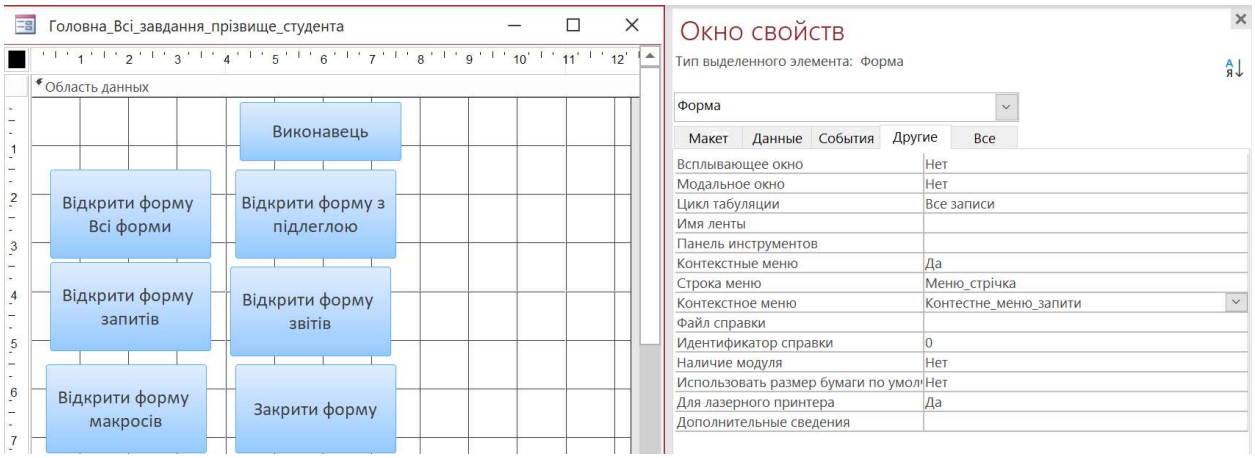



Рис. 12.8. Зв'язування макроса меню з контекстним меню відповідної форми

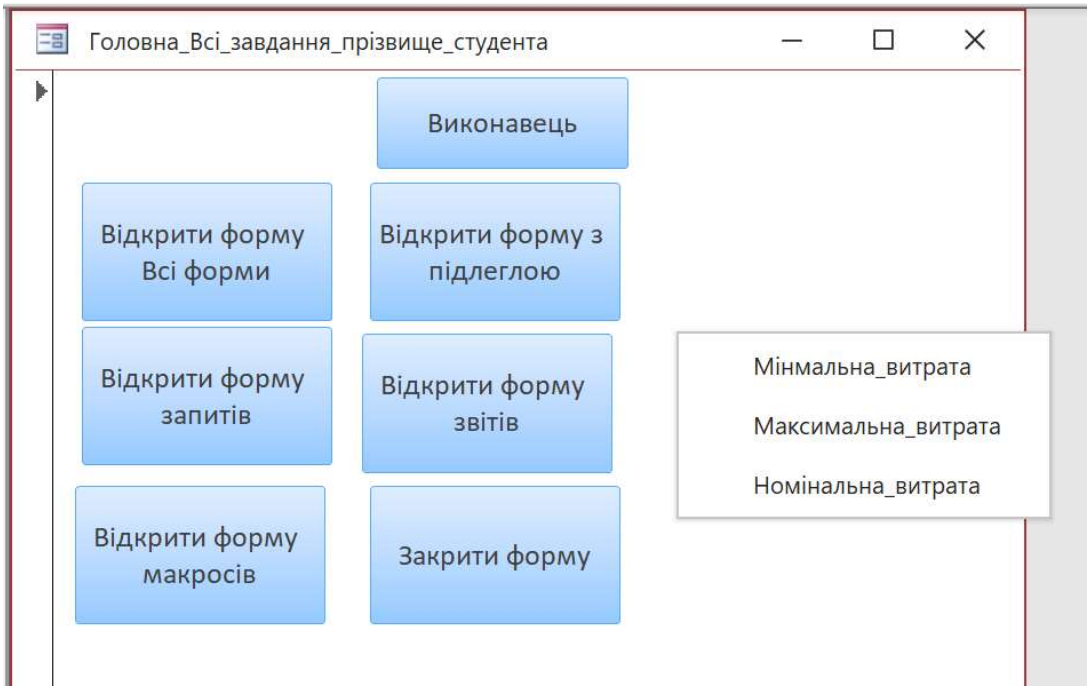



Рис. 12.9. Вигляд форми з відкритим контекстним меню

## 12.3. Створення гарячих клавіш

Гарячі клавіші використовуються як альтернатива для швидкого виконання дій, які передбачені відповідними опціями меню. Розглянемо створення гарячих клавіш на прикладі. Створимо гарячі клавіші для виклику підопцій меню **Всі_таблиці**.

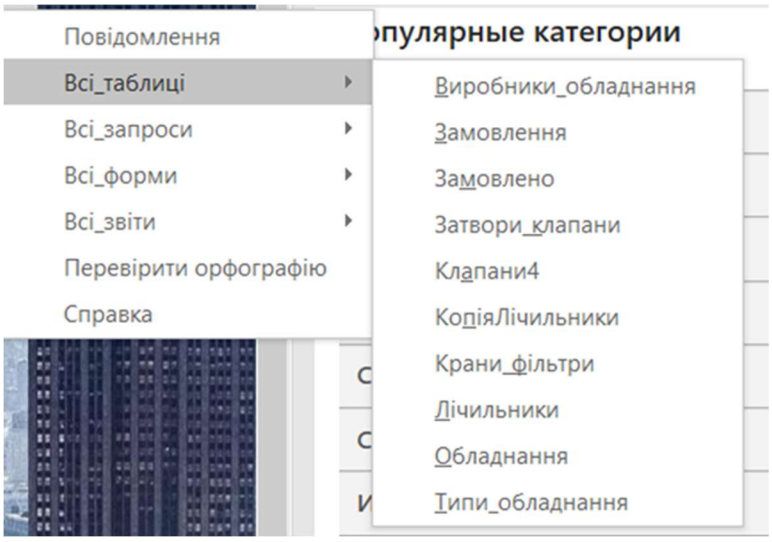



Рис. 12.10. Підопції меню Всі_таблиці для реалізації гарячих клавіш

При наведенні курсору на опцыю меню **Всі_таблиці** відкривається меню з усіма таблицями даної бази даних. Можна помітити, що одна літера у кожному рядку меню є підкресленою.

Наприклад, після виклику розробленого контекстного меню користувача, наведемо курсор на опцію **Всі_таблиці** (у цьому випадку повинна бути обрана українська розкладка клавіатури) і натиснемо літеру «О».

Відкривається таблиця «Обладнання»

Обладнання

| Код_обладн | Код_типу | Ціна | Код_вироб | На_складі | Очікується | Мін_запас | По |
|---|---|---|---|---|---|---|---|
| 1915_13_03 | 7 | 51,00 грн. | 7 | 87 | 12.03.2879 | 10 | |
| 1915_13_06 | 7 | 51,00 грн. | 7 | 32 | 03.02.2860 | 18 | |
| 1915_13_08 | 7 | 51,00 грн. | 7 | 34 | 03.02.2860 | 15 | |
| 1915_13_10 | 7 | 51,00 грн. | 7 | 65 | 30.12.2859 | 16 | |
| 1915_13_2,5 | 7 | 51,00 грн. | 7 | 21 | 03.02.2860 | 13 | |
| 1915_19_03 | 7 | 78,00 грн. | 7 | 89 | 30.12.2859 | 15 | |
| 1915_19_06 | 7 | 78,00 грн. | 7 | 32 | 03.02.1900 | 9 | |
| 1915_19_08 | 7 | 78,00 грн. | 7 | 45 | 30.12.2859 | 18 | |
| 1915_19_10 | 7 | 78,00 грн. | 7 | 64 | 12.04.2980 | 10 | |
| 1915_19_2,5 | 7 | 78,00 грн. | 7 | 12 | 03.02.2860 | 12 | |
| 1915_25_03 | 7 | 139,20 грн. | 7 | 56 | 30.12.2859 | 14 | |
| 1915_25_06 | 7 | 139,20 грн. | 7 | 34 | 03.02.2860 | 13 | |
| 1915_25_08 | 7 | 139,20 грн. | 7 | 76 | 12.04.2980 | 10 | |
| 1915_25_10 | 7 | 139,20 грн. | 7 | 54 | 30.12.1899 | 5 | |
| 1915_25_2,5 | 7 | 139,20 грн. | 7 | 85 | 30.12.2859 | 16 | |
| 1915_31_03 | 7 | 333,60 грн. | 7 | 73 | 30.12.2859 | 18 | |
| 1915_31_06 | 7 | 333,60 грн. | 7 | 85 | 12.04.2980 | 10 | |
| 1915_31_08 | 7 | 333,60 грн. | 7 | 70 | 30.12.2859 | 18 | |
| 1915_31_10 | 7 | 333,60 грн. | 7 | 64 | 30.12.2859 | 12 | |

Рис. 12.11. Результат виконання опції меню Всі_таблиці для при виборі гарячої клавіші «О»

Для реалізації гарячої клавіші , потрібно у відповідному макросі «Всі_таблиці», який складається з вкладених макросів, біля необхідної для нас букви поставити знак «&» амперсант.

**Вложенный макрос:** &Обладнання

**ОткрытьТаблицу**

Имя таблицы Обладнання

Режим Таблица

Режим данных Изменение

**Конец вложенного макроса**

Для того щоб для всіх об’єктів (таблиці, запити, форми, звіти) бази даних було доступним створене контекстне меню, необхідно зайти в параметри Access

і в контекстному меню за замовчуванням набрати ім’я нашого макросу **Контекстне меню** рис. 12.12.

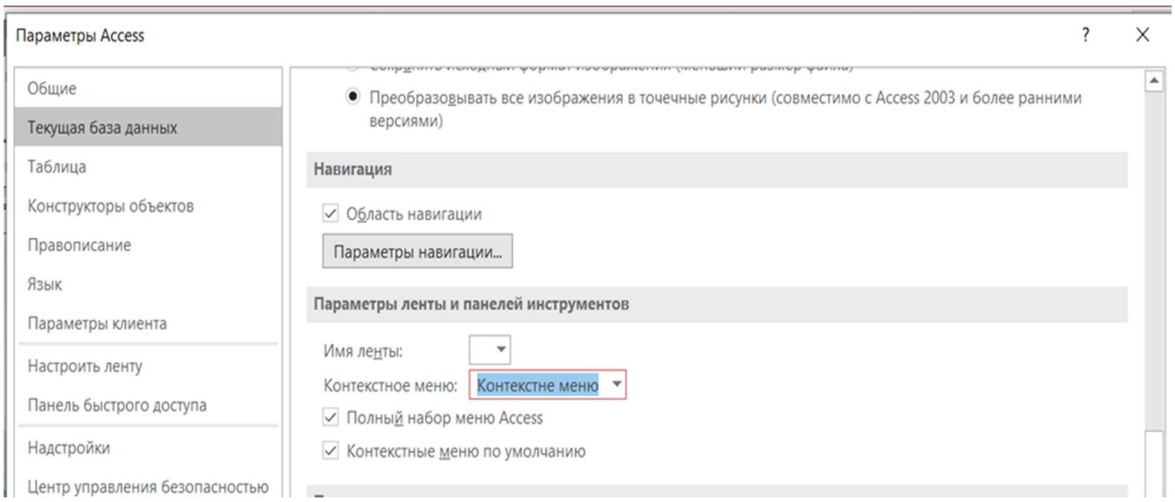



Рис. 12.12.

## 12.4. Завдання до комп'ютерного практикуму

1. Створити головне меню у формі **Головна_форма_прізвище_студента_гр_ПМ-ХХ**, до якого включити всі створені у базі об’єкти (таблиці, форми, запити та звіти). Пункти головного меню назвати **Всі_форми, Всі_макроси, Всі_запити та Всі_звіти**. Кожен з пунктів головного меню повинен складатися з підменю, пункти яких будуть відкривати відповідні об’єкти класів (наприклад, опція головного меню **Всі_форми** складається з підопцій, за допомогою яких можна запустити на виконання будь-яку із створених форм). Наприклад,

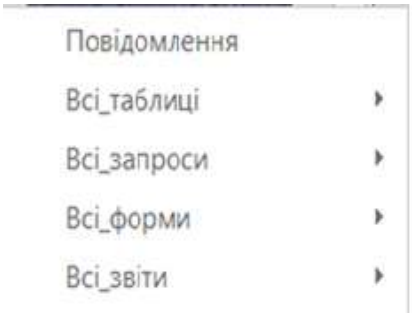



Рис. 12.13.

Замість назви меню **Повідомлення** використати Робота_з_об’єктами_прізвище_студента. Опцію меню **Всі_таблиці** замінити на опцію **Всі_макроси** (мають викликатися макроси індивідуального завдання КП 8), опцією меню **Всі_запити** мають викликатися завдання КП 6).

2. Створити контекстне меню у формі **Головна_Всі_завдання_прізвище_студента** . Опціями контекстного меню мають стати запити або приклади індивідуального завдання комп’ютерного практикуму 6.
3. Створити гарячі клавіші для виклику опцій меню **Таблиці_ вар_№** (тільки для таблиць індивідуального завдання).
4. Створити окреме контекстне меню у формі **Пошук** для відкриття форм **Пошук_лічильників** і **Пошук_таблиці_інд_завдання_КП**_9.
5. Для цьому потрібно створити головний макрос **Контекстне пошук вар.№** і макрос **Пошук_вар_№,** який і буде відкривати форми **Пошук_лічильників** та **Пошук_таблиці_інд_завдання_КП_9.**
6. Скласти протокол з детальним описом розробки завдань пунктів 1-5 завдання комп'ютерного практикуму 12. Дати письмові відповіді на контрольні запитання.

## 12.5. Контрольні запитання

1. Як створити меню, яке відкривається лише при активізації відповідого звіту? Наведіть на прикладі послідовність створення такого меню для звіту

2. Як створити окрему панель інструментів з власноруч вибраними піктограмками?

3. Як створити попереджуючі повідомлення при виконанні пунктів меню?

4. Які особливості створення контекстного меню?

5. Як переглянути та відредагувати контекстні меню, що створені розробником додатку бази даних ?

6. Як створити контекстне меню, яке викликалось би з декількох об'єктів бази даних?

## Список рекомендованої і використаної літератури


1. Дейт К. Введение в системы баз данных. – М.: Издательский дом «Вильямс», 8-е издание, 2019.-1328 с.
2. Коннолли Т., Бегг К., Базы данных: проектирование, реализация и сопровождение. Теория и практика, 3-е изд.: Пер. с англ. – М.: Издательский дом «Вильямс», 2017.- 1440 с.
3. https://support.microsoft.com/uk-ua/office .
4. Бекаревич Ю.Б. Самоучитель Access 2013. - СПб.: BHV – Санкт-Петербург, 2014 . – 464 с.
5. https://www.bestprog.net/uk/2018/02/17/the-datagridview-control_ua/
6. Пасічник В.В., Резніченко В.А. Організація баз даних та знань. – К.: Видавнича група BHV, 2006.- 384 с.
7. Харитонова И.А., Михеева В.Д. Microsoft Access 2003: разработка приложений.- СПб.: BHV – Санкт-Петербург, 2004 . – 1072 с.
8. Праг К., Ирвин М. Access 2000. Библия пользователя. Пер. с англ. – М.: Издательский дом «Вильямс», 2004.- 1040 с.
9. Мак-Федрис П. Формы, отчеты и запросы в Microsoft Access 2000: Пер. с англ. – М.: Издательский дом «Вильямс», 2003.- 416 с.