МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»
КАФЕДРА ПРОГРАМНИХ І КОМП’ЮТЕРНО-ІНТЕГРОВАНИХ ТЕХНОЛОГІЙ

МЕТОДИЧНІ ВКАЗІВКИ З ДИСЦИПЛІНИ ОСНОВИ БАЗ ДАНИХ.

(Теоретична частина)

Для студентів інституту штучного інтелекту та робототехніки

Перший (бакалаврський) рівень вищої освіти

Спеціальність: 151 – Автоматизація та комп'ютерно-інтегровані технології

Освітньо-професійна програма: Комп'ютерні технології автоматизації.

Одеса 2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»
КАФЕДРА ПРОГРАМНИХ І КОМП'ЮТЕРНО-ІНТЕГРОВАНИХ ТЕХНОЛОГІЙ

МЕТОДИЧНІ ВКАЗІВКИ З ДИСЦИПЛІНИ ОСНОВИ БАЗ ДАНИХ.

(Теоретична частина)

Для студентів інституту штучного інтелекту та робототехніки

Перший (бакалаврський) рівень вищої освіти

Спеціальність: 151 – Автоматизація та комп'ютерно-інтегровані технології

Освітньо-професійна програма: Комп'ютерні технології автоматизації.

Затверджено на засіданні
кафедри програмних і комп'ютерно-інтегрованих технологій
Протокол № 7 від 26.01.2022 р.

Одеса 2022

Методичні вказівки з дисципліни Основи баз даних. (Теретична частина): для студ. напряму 151 «Автоматизація та комп'ютерно-інтегровані технології» денної та заочної форм навчань./ Укл. Лисюк Г.П. – Одеса: ОП, 2022. – 101 с.

ЗМІСТ

1 ОСНОВНІ ПОНЯТТЯ БАЗ ДАНИХ. РЕЛЯЦІЙНІ БАЗИ. КЛЮЧІ НОРМАЛІЗАЦІЯ, ЗВ'ЯЗКИ. АРХІТЕКТУРА ACCESS.

1.1 ОСНОВНІ ПОНЯТТЯ.

Інформаційна система - це сукупність програмно-апаратних засобів, способів і людей, які забезпечують збирання, зберігання, обробку та видачу інформації для вирішення поставлених завдань.

Бази даних є сучасною формою організації, зберігання та доступу до інформації. Прикладами великих інформаційних систем є банківські системи, системи замовлень залізничних квитків тощо.

База даних – це інтегрована сукупність структурованих та взаємопов'язаних даних, організована за певними правилами, що передбачають загальні принципи опису, зберігання та обробки даних. Зазвичай, база даних створюється для предметної області.

Предметна область – це частина реального світу, що підлягає вивченню з метою створення бази для автоматизації процесу управління.

Структурування – запровадження угод про засоби представлення даних.

Самі собою бази даних не становили б інтересу, якби не було систем управління базами даних.

До роботи з даними використовуються системи управління базами даних (СУБД). СУБД - це комплекс програмних та мовних засобів, необхідних для створення баз даних, підтримки їх у актуальному стані та організації пошуку необхідної інформації.

Загалом СУБД - це система, що дозволяє створювати бази даних та маніпулювати відомостями з них. А здійснює цей доступ до даних СУБД у вигляді спеціальної мови – SQL(structured query language).

SQL - мова структурованих запитів, основним завданням якого є надання простого способу зчитування та запису інформації до бази даних.

За характером використання СУБД ділять на однокористувацькі (призначені для створення та використання БД на персональному комп'ютері) і розраховані на багато користувачів (призначені для роботи з єдиною БД декількох комп'ютерів, об'єднаних у локальні мережі).

Набори принципів, що визначають організацію логічної структури зберігання даних у основі, називаються моделями даних.

Існують основні моделі даних: списки (плоські таблиці), реляційні бази даних, ієрархічні та мережеві структури.

Протягом багатьох років переважно використовувалися плоскі таблиці (плоскі БД) типу списків Excel.

Ієрархічна структура бази даних

Це деревоподібна структура представлення інформації. Її особливість у тому, що кожен вузол на нижчому рівні має зв'язок тільки з одним вузлом на вищому рівні. Приклад фрагмента ієрархічної структури бази даних "Інститут":

Зі структури зрозуміло, що на одній кафедрі може працювати кілька викладачів. Такий зв'язок називається "один до багатьох" (одна кафедра – багато викладачів). Але якщо ми спробуємо додати до цієї структури групи студентів, то нам знадобиться зв'язок "багато хто до багатьох":

Один викладач може працювати з багатьма групами, а одна група може навчатися у багатьох викладачів, а такого зв'язку в ієрархічній структурі бути не може (бо зв'язок може бути тільки з одним вузлом на вищому рівні). Це основний недолік подібної структури бази даних.

Мережева структура бази даних

По суті це розширення ієрархічної структури. Все те ж саме, але існує зв'язок "многим-багатьом ". Мережева структура бази даних дозволяє нам додати групи до нашого прикладу. Недоліком мережі є складність розробки серйозних додатків.

Реляційна структура бази даних

Нині найбільшого поширення розробки БД отримали реляційні моделі даних. Реляційна модель даних є сукупністю найпростіших двовимірних таблиць – відносин (англ. relation), тобто. Найпростіша двовимірна таблиця визначається як відношення (безліч однотипних записів об'єднаних однією темою).

Від терміна relation (ставлення) походить назва реляційна модель даних. У реляційних БД використовується кілька двомірних таблиць, у яких рядки називаються записами, а стовпці полями, між записами яких встановлюються зв'язки. Цей спосіб організації даних дозволяє дані (запису) в одній таблиці пов'язувати з даними (записами) в інших таблицях через унікальні ідентифікатори (ключі) або ключові поля.

Об'єктами обробки реляційної БД є такі інформаційні одиниці: полі, запис, таблиця. Поле – елементарна одиниця логічної організації даних, що відповідає одному реквізиту інформаційного об'єкта (стовпець реляційної таблиці). Запис – сукупність логічно пов'язаних полів (узагальнений рядок реляційної таблиці). Примірник запису – окрема реалізація запису, що містить конкретні значення її полів (конкретний рядок реляційної таблиці). Таблиця - задана структура полів, що складається з кінцевого набору однотипних записів.

1.2 КЛЮЧІ РЕЛЯЦІЙНИХ БД.

Основними об'єктами будь-якої бази є її таблиці. Таблиці бази даних створюються таким чином, щоб кожна з них містила інформацію про один інформаційний об'єкт. Після створення різних таблиць між цими таблицями мають бути встановлені реляційні зв'язки. Установка таких зв'язків уможливлює виконання одночасної обробки даних з декількох таблиць. Для встановлення зв'язків зазвичай використовують ключові поля таблиць.

Ключ – це стовпець (можливо кілька стовпців), який додається до таблиці і дозволяє встановити зв'язок із записами в іншій таблиці.

Існують ключі двох типів: первинні та вторинні чи зовнішні.

Первинний ключ – це одне або кілька полів (стовпців), комбінація значень яких однозначно визначає кожен запис у таблиці. Первинний ключ не допускає значень Null і повинен мати унікальний індекс. Первинний ключ використовується для зв'язування таблиці із зовнішніми ключами в інших таблицях.

Зовнішній (вторинний) ключ – це одне або кілька полів (стовпців) у таблиці, що містять посилання на полі чи поля первинного ключа в іншій таблиці. Зовнішній ключ визначає спосіб поєднання таблиць.

Зовнішній ключ, як і первинний ключ, може бути комбінацією стовпців. Насправді зовнішній ключ завжди буде складовим, якщо він посилається на складовий первинний ключ іншої таблиці. Кількість стовпців та їх типи даних у первинному та зовнішньому ключах повинні збігатися.

Якщо таблиця пов'язані з декількома іншими таблицями, може мати кілька зовнішніх ключів.

З двох логічно пов'язаних таблиць одну називають таблицею первинного ключа, чи головною таблицею, іншу – таблицею вторинного (зовнішнього) ключа, чи підлеглою таблицею. СУБД дозволяють зіставити родинні записи з обох таблиць і вивести їх у формі, звіті або запиті.

Існує три типи первинних ключів: ключові поля лічильника (лічильник), простий ключ та складовий ключ.

Поле лічильника (Тип даних "Лічильник"). Тип даних поля в базі даних, в якому для кожної запису, що додається в таблицю, в поле автоматично заноситься унікальне числове значення.

Простий ключ. Якщо поле містить унікальні значення, такі як коди або інвентарні номери, це поле можна визначити, як первинний ключ. Як ключ можна визначити будь-яке поле, що містить дані, якщо це поле не містить повторювані значення або значення Null.

Складовий ключ. Якщо неможливо гарантувати унікальність значень кожного поля, можна створити ключ, що складається з декількох полів. Найчастіше така ситуація виникає при створенні таблиці, що використовується для зв'язування двох таблиць багатьом.

Необхідно вкотре помітити, що у полі первинного ключа мають бути лише унікальні значення у кожному рядку таблиці, тобто. збіг не допускається на відміну поля вторинного чи зовнішнього ключа, де збіг значень у рядках таблиці допускается.

Якщо виникають труднощі з вибором відповідного типу первинного ключа, то як ключ доцільно вибрати поле лічильника.

1.3 НОРМАЛІЗАЦІЯ БД.

При створенні додатків баз даних найбільш важливим етапом є конструювання БД (визначення структур таблиць, зв'язків між ними і т.д.). Помилки у структурі даних важко, а частіше взагалі неможливо виправити програмним шляхом. Що краще структура даних, то легше програмувати БД. Теорія проектування БД містить концепцію нормальних форм, призначених оптимізації структури БД.

Нормальні форми - це лінійна послідовність правил, що застосовуються до БД, причому що стоїть номер нормальної форми, тим досконаліше структура БД.

Нормалізація - це багатоступінчастий процес, у якому таблиці БД організуються, роз'єднуються і дані упорядковуються. Концепцію нормалізації вперше представив О.Ф. Кодд (E.F. Codd) у 1970-і роки. Завдання нормалізації залишається тим самим і сьогодні: усунути з БД деякі небажані характеристики. Зокрема, ставиться завдання усунути деякі види надмірності даних та завдяки цьому уникнути аномалій при зміні даних.

Аномалії зміни даних - це складності при операціях вставки, зміни та видалення даних, що виникають через структуру БД. Хоча існує багато рівнів, зазвичай достатньо виконати нормалізацію до третьої нормальної форми (3НФ).

Принципи нормалізації:

- У кожній таблиці БД не повинно бути полів, що повторюються;
- У кожній таблиці має бути унікальний ідентифікатор (первинний ключ);
- Кожному значенню первинного ключа повинна відповідати достатня інформація про тип сутності або об'єкт таблиці (наприклад, інформація про успішність, групу або студентів);
- Зміна значень у полях таблиці не повинна впливати на інформацію в інших полях (крім змін у полях ключа).

Розбивати треба тому, що з використанням універсального ставлення виникає кілька проблем:

1. Надмірність. Дані багатьох стовпців багаторазово повторюються.

2. Потенційна суперечливість (аномалії поновлення). Внаслідок надмірності можна оновити адресу постачальника в одному рядку, залишаючи її незмінною в інших. Отже, при оновленнях необхідно переглядати всю таблицю для знаходження та зміни всіх відповідних рядків.

3. Аномалії включення.

4. Аномалії видалення.

Розглянемо приклад нормалізації БД управління доставкою замовлень. Невпорядкована БД " Продажу " складалася б з однієї таблиці " Продажу " виглядала б так:

Клієнт	Код товару	Найменування товару	Кількість	Ціна	Усього
1	121,333,444	Лампа, Ножиці, Парасолька	5,2,8	2,4,10	10,8,80

У таблиці кожен запис містить відомості про кілька замовлень одного клієнта. Оскільки стовпець з даними про товар містить дуже багато даних, отримати впорядковану інформацію з цієї таблиці складно (наприклад, скласти звіт про сумарні закупівлі з різних видів товарів).

Перша нормальна форма (1НФ)

Перша нормальна форма(1НФ) визначає атомарність всіх даних, що у стовпцях. Слово "атом" походить від латинського "atomis", що буквально означає "не підлягає поділу". 1НФ задає існування в кожній позиції, що визначається рядком та стовпцем, лише одного значення, а не масиву чи списку значень. Переваги цієї вимоги є очевидними: якщо в одному стовпці зберігаються списки значень, то не існує простого способу маніпулювати цими значеннями. Звичайно, при цьому збільшується кількість записів у таблиці.

Таблиця знаходиться в першій нормальній формі (1НФ) тоді і тільки тоді, коли жодна з її рядків не містить у своєму полі більше одного значення і жодне з її ключових полів не порожнє.

Виконаємо нормалізацію БД " Продажі" до 1НФ:

Клієнт	Код товару	Найменування товару	Кількість	Ціна	Усього
1	121	Лампа	5	2	10
1	333	Ножиці	2	4	8
1	444	Парасолька	8	10	80

Друга нормальна форма (2НФ)

До Другої нормальної форми (2НФ) можна перейти від таблиці, яка відповідає 1НФ. Додатково має виконуватися така умова: кожне неключове поле має залежати від первинного ключа.

Таблиця знаходиться у другій нормальній формі (2НФ), якщо вона задовольняє визначенню 1НФ і всі її поля, що не входять до первинного ключа, пов'язані повною функціональною залежністю з первинним ключем.

Ця форма застосовується до таблиць зі складовими ключами. Таблиця, у якій первинний ключ включає лише одне поле, завжди знаходиться у 2НФ.

Виконаємо нормалізацію БД " Продажі " до 2НФ. Усі відомості, не пов'язані з окремими замовленнями, виділимо окрему таблицю. У підсумку отримаємо замість однієї таблиці "Продаж" дві - таблицю "Замовлення":

Клієнт	Код товару	Кількість	Всього
1	121	5	10
1	333	2	8
1	444	8	80

та таблицю "Товари":

Код товару	Найменування товару	Ціна
121	Лампа	2
333	Ножиці	4
444	Парасолька	10

Таким чином, вид товару зберігається лише в одній таблиці. Слід звернути увагу, що при нормалізації інформація не втрачається.

Третя нормальна форма (3НФ)

Таблиця знаходиться у третій нормальній формі (3НФ), якщо вона задовольняє визначенню 2НФ і не одне з її неключових полів не залежить функціонально від будь-якого іншого неключового поля.

Вважається, що таблиця відповідає Третьій нормальній формі (3НФ), якщо вона відповідає 2НФ і всі ключові стовпці взаємно незалежні. Стовпець, значення якого виходять обчисленням на основі даних з інших стовпців, є одним із прикладів залежності.

Виконаємо нормалізацію БД "Продажі" до 3НФ. Для цього слід видалити з таблиці "Замовлення" стовпець "Усього". Значення в цьому стовпці не залежать від жодного ключа і можуть бути обчислені за формулою ("Ціна")*("Кількість").

Таблиця "Замовлення"

Клієнт	Код товару	Кількість
1	121	5
1	333	2
1	444	8

Таблиця "Товари":

Код товару	Найменування товару	Ціна
121	Лампа	2
333	Ножиці	4
444	Парасолька	10

Таким чином, отримано БД "Продажі" з оптимальною структурою.

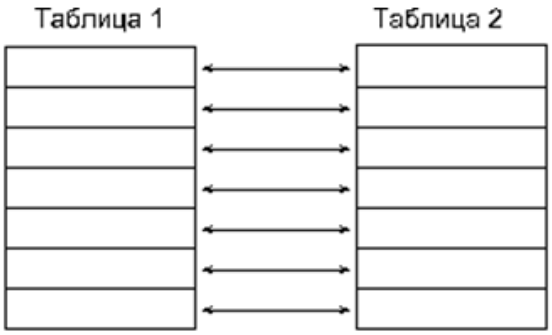
1.4 ВИДИ ЛОГІЧНОГО ЗВ'ЯЗКУ. ЦІЛІСНІСТЬ БАЗИ ДАНИХ.

Реляційна база даних є безліч взаємозалежних таблиць, кожна з яких містить інформацію про об'єкти певного типу. Кожен рядок таблиці включає дані про один об'єкт (наприклад, клієнт, автомобіль, документ), а стовпці таблиці містять різні характеристики цих об'єктів - атрибути (наприклад, найменування та адреси клієнтів, марки та ціни автомобілів). Рядки таблиці називаються записами; всі записи мають однакову структуру – вони з полів, у яких зберігаються атрибути об'єкта. Кожне поле запису містить одну характеристику об'єкта і має певний тип даних (наприклад, текстовий рядок, число, дата). Усі записи мають одні й самі поля, але різні значення атрибутів.

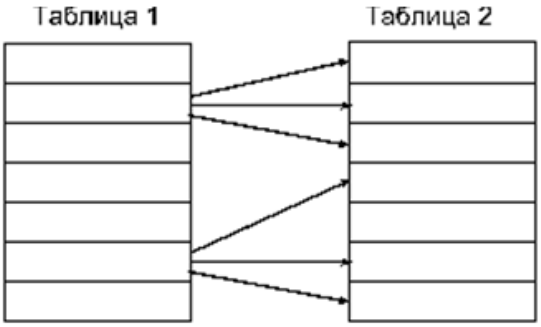
Перш ніж заносити дані таблиці потрібно визначити структуру цих таблиць. Під цим розуміється як опис найменувань і типів полів, а й інших характеристик (наприклад, формат, критерії перевірки введених даних). Крім опису структури таблиць, зазвичай

задаються зв'язки між таблицями. Зв'язки в реляційних базах даних визначаються за збігом значень полів у різних таблицях:

При відношенні "один-до-одному" кожного запису однієї таблиці ставиться у відповідність один запис іншої таблиці. Прикладом можуть бути таблиця співробітників та таблиця їх адрес. Цей тип зв'язку використовують не дуже часто, оскільки такі дані можуть бути розміщені в одну таблицю.



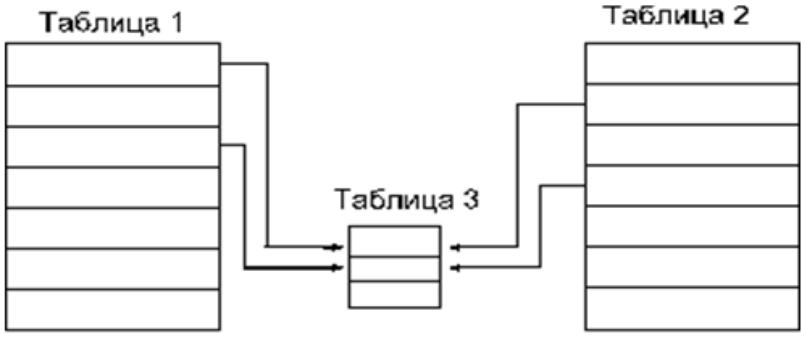
Відношення "один-багатьом" у базах даних зустрічається найчастіше. У цьому випадку одного запису однієї таблиці ставиться у відповідність кілька записів в іншій таблиці. Однак, кожен запис другої таблиці не може мати більше одного відповідного запису першої таблиці.



Прикладом можуть бути таблиця співробітників і таблиця їх заробітної плати по місяцях (один співробітник отримує протягом року кілька разів зарплату).

- "багато -к-одному" – безлічі записів з однієї таблиці відповідає один запис в іншій таблиці;

При відношенні "многим-багатьом" одного запису однієї таблиці може відповідати кілька записів другої таблиці, а одного запису другої таблиці - кілька записів першої.) таблиці, що складається з первинних ключів двох перших таблиць.



Наприклад, такий тип зв'язку реалізується під час створення бібліотечного каталогу. Один автор може написати кілька книг, але й одна книга може належати кількома авторам.

Таким чином, мають існувати таблиця авторів, таблиця книг та третя таблиця з назвами книг та іменами авторів.

Після створення різних таблиць, що містять дані, необхідно продумати, яким чином об'єднати ці дані при їх витягуванні з БД. Насамперед необхідно створити зв'язок між таблицями, які встановлюють відносини між збігаються значеннями у ключових полях (зазвичай між однойменними полями різних таблиць) . Наприклад, у розглянутому вище прикладі зв'язок між таблицею "Замовлення" та "Товари" здійснюється по полю "Код товару". Після створення зв'язків між таблицями стає можливим створення запитів, форм та звітів, у яких виводяться дані з кількох таблиць відразу. Завдяки використанню зв'язків між таблицями вдається значно зменшити обсяг будь-якої бази даних, особливо у випадках, коли інформація повторюється.

Схема бази даних (чи навіть схема даних) є графічним чином БД. У ній визначаються та запам'ятовуються зв'язки між таблицями. Це дозволяє Access автоматично використовувати зв'язку під час конструювання форм, запитів, звітів. Схема даних відображається у спеціальному вікні Схема даних, де таблиці представлені списками полів, а зв'язки – лініями між полями у зв'язаних таблицях (рис. 1).

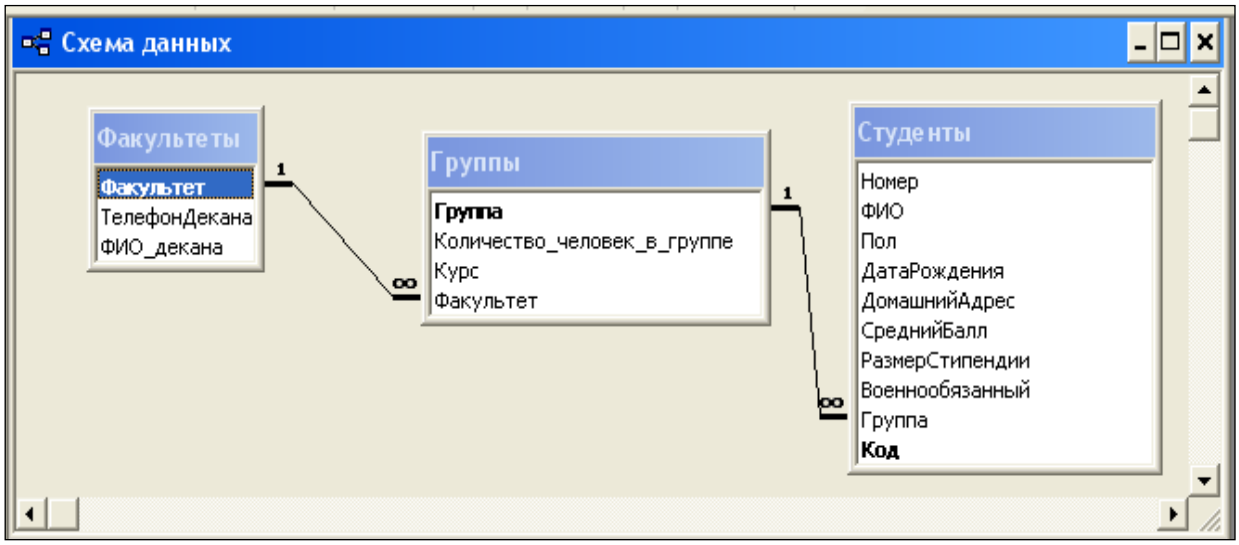


Рис. 1. Схема БД Студент

Створення схеми даних передбачає наявність нормалізованих таблиць, зв'язування яких у вікні Схема даних. Створення схеми даних починається з розміщення у вікні всіх таблиць, які мають бути включені до схеми. Далі можна приступати до визначення попарних зв'язків між ними.

Встановлюючи зв'язку між парою таблиць, треба виділити в головній таблиці унікальне ключове поле (ПК у кожній таблиці відображається жирним шрифтом), яким встановлюється зв'язок. Далі, при натиснутій лівій кнопці миші це поле перетягується на відповідне поле підпорядкованої таблиці. Якщо встановлюється зв'язок за складеним ключем, необхідно виділити всі поля, що входять до складової ключ головної таблиці, і перетягнути їх на одне з полів зв'язку в підпорядкованій таблиці.

Після встановлення зв'язку відкривається діалогове вікно Зміна зв'язку, у якому ключового поля головної таблиці визначається поле зв'язку підпорядкованої таблиці. Для кожного поля складеного ключа головної таблиці зв'язок з полем підпорядкованої таблиці має бути встановлений окремим рядком. Крім того, у вікні Зміна зв'язку для кожного зв'язку можна встановити параметр Забезпечення цілісності даних, після чого встановлюються опції Каскадне оновлення пов'язаних полів та Каскадне видалення пов'язаних записів. У цьому Access автоматично встановить тип зв'язку 1:М (у схемі позначається як 1:∞). Якщо

таблиці містять дані, що не відповідають вимогам цілісності, зв'язок 1:М не буде встановлений, і Access у цьому випадку виводить відповідне повідомлення.

Цілісність бази даних

Цілісність даних означає систему правил, що використовуються для підтримки зв'язків між записами у зв'язаних таблицях, а також для захисту від випадкового видалення або зміни пов'язаних даних. Правила цілісності повинні зберігатися разом з базою даних та дотримуватися на глобальному рівні.

Функціонування цілісності даних грунтується на ключових полях. У РБД, перш за все, має забезпечуватися вимога цілісності даних за посиланнями, яка полягає в тому, що для кожного значення зовнішнього ключа (ВК) підлеглого відношення в основному (головному) відношенні має знайти кортеж (запис) з таким самим значенням первинного ключа (ПК)). В іншому випадку значення ВК має містити значення Null, тобто бути невизначеним (ні на що не вказувати). Це обмеження РСУБД, як правило, підтримується автоматично.

Якщо модель даних розроблена відповідно до вимог нормалізації таблиць, то для того, щоб забезпечити цілісність, робота з даними повинна проводитися з урахуванням нижченаведених правил:

• неможливо ввести у пов'язане поле підпорядкованої таблиці значення, яке відсутнє у зв'язаному полі головної таблиці;

• не допускається видалення запису з головної таблиці, якщо існують пов'язані з нею записи підпорядкованої таблиці;

• неможливо змінити значення ключового поля в головній таблиці, якщо є записи в підпорядкованій таблиці, пов'язані з даною таблицею.

Встановлення між двома таблицями зв'язку типу 1:1 або 1:М та завдання параметрів цілісності даних можливе за таких умов:

• поля, що зв'язуються, мають однаковий тип даних, при цьому імена полів можуть бути різні;

• таблиці зберігаються у одній БД;

• головна таблиця пов'язується з підлеглою за первинним простим чи складовим ключем головної таблиці.

Access не дозволяє створювати зв'язки з параметрами забезпечення цілісності даних, якщо раніше введені таблиці дані не відповідають вимогам цілісності. При введенні некоректних даних пов'язані таблиці Access виводить відповідне повідомлення.

Кожен зв'язок має низку властивостей. Якщо для вибраного зв'язку забезпечується підтримка цілісності, можна встановити опції *Каскадне оновлення пов'язаних полів* та *Каскадне видалення пов'язаних записів*. Перша опція дозволяє при зміні ПК в основній таблиці автоматично змінити ВК у всіх пов'язаних таблицях. Друга опція дозволяє видалити запис з основної таблиці та всі пов'язані з нею записи у підпорядкованій таблиці. При спробі внесення змін до даних, які можуть порушити умову цілісності, Access відображає діалогове вікно, що перешкоджає внесенню цих змін.

1.5 ОСНОВИ РОБОТИ ІЗ СУБД MICROSOFT ACCESS.

Створення БД починається із проектування.

Етапи проектування БД:

- Вивчення предметної області;

- аналіз даних (сутностей та його атрибутів);

- визначення відносин між сутностями та визначення первинних та вторинних (зовнішніх) ключів.

У процесі проектування визначається структура реляційної БД (склад таблиць, їх структура та логічні зв'язки). Структура таблиці визначається складом стовпців, типом даних та розмірами стовпців, ключами таблиці.

Microsoft Access - це СУБД, що пропонує широкий діапазон засобів для зберігання інформації та ефективного управління цією інформацією. База даних у Access - це один файл, що містить таблиці, запити та інші об'єкти БД. Робота з будь-якими об'єктами виконується у вікні База даних. На лівій панелі цього вікна знаходяться елементи керування, щоб викликати всі типи об'єктів.

Панель "Об'єкти":

1. Таблиця – двомірні таблиці, що використовуються зберігання даних у реляційних базах даних. Дані зберігаються у записах, які з окремих полів. Кожна таблиця містить інформацію про сутності певного типу (наприклад, студентів).

2. Запит - засіб відбору даних, які відповідають певним умовам. За допомогою запитів можна вибрати з бази даних лише необхідну інформацію.

3. Форма – засіб, що дозволяє спростити процес введення чи зміни даних у таблицях БД, що забезпечує введення даних персоналом невисокої кваліфікації.

4. Звіт - засіб, який дозволяє витягти з бази потрібну інформацію та подати її у вигляді, зручному для сприйняття, а також підготувати для роздруківки звіт, який оформлений відповідним чином.

5. Сторінки - сторінки доступу до даних є спеціальною Web-сторінкою, призначеною для перегляду та роботи через Інтернет або інтрасетю з даними, що зберігаються в базах даних Microsoft Access або БД MS SQL Server.

6. Макрос - набір макрокоманд, створюваний користувачем автоматизації виконання конкретних операцій.

7. Модуль - об'єкт, що містить програми мовою Visual Basic, які застосовуються в деяких випадках для обробки даних.

2 СТВОРЕННЯ ТАБЛИЦЬ В ACCESS.

2.1 РЕЖИМИ РОБОТИ З ОДИННИМИ ТАБЛИЦЯМИ.

У Access є режими роботи з таблицями: режим Таблиці, режим Конструктора.

У режимі Таблиці здійснюється робота з даними, що знаходяться в одиночній таблиці БД: перегляд, редагування, додавання, сортування тощо.

У режимі Конструктора створюється чи модифікується структура таблиці, тобто задаються імена полів, їх типи, опис, властивості та інші параметри.

Існує також додатковий режим - Попередній перегляд, який дозволяє побачити розташування даних на аркуші перед здійсненням друку таблиці.

Створення таблиці БД і двох етапів.

На першому етапі визначається її структура: склад та імена полів, ключі, послідовність розміщення полів у таблиці, тип даних кожного поля, властивості полів (таких, як розмір поля, формат, індексоване поле та ін.).

На другому етапі проводиться ідентифікація записів таблиці та заповнення їх даними.

2.2 СТВОРЕННЯ ТАБЛИЦЬ.

2.2.1 СТВОРЕННЯ СТРУКТУРИ ТАБЛИЦІ

Створення таблиці починається з визначення її структури (конструювання таблиці) як Конструктора таблиць. Режим Конструктора дозволяє користувачеві вказати параметри всіх елементів структури таблиці.

Створення структури таблиці – це багатокроковий процес. Для швидкого конструювання таблиці необхідно виконати такі дії:

1. Відкрити вікно Конструктора.

2. Ввести імена полів, вказати тип даних та їх опис.

3. Для кожного поля введіть властивості.

4. Встановити первинний ключ.

5. У разі потреби створити індексовані поля.

6. Зберегти таблицю із створеною структурою.

Присвоєння імені полям таблиці

Ім'я поля дозволяє Access ідентифікувати певне поле. Воно має бути змістовним та коротким, тому що довгі імена полів незручні для обробки. При введенні імені необхідно дотримуватися таких правил.

- Ім'я поля не повинне починатися з пробілу.

– Ім'я поля може містити від 1 до 64 символів з урахуванням пробілів.

- В іменах полів можна використовувати і малі, і великі літери.

– Імена полів можуть включати літери, цифри та спеціальні символи. Вони можуть містити крапку (.), знак вигуку (!), квадратні дужки ([ ]) і знак апострофа (').

Ім'я поля можна змінити або редагувати в будь-який момент (навіть якщо воно вже знаходиться в таблиці та містить дані).

Завдання типу даних

Після завдання імені поля потрібно вибрати тип даних. У табл. 2.1 перераховано основні типи даних.

Таблиця 2.1

Тип даних	Опис	Розмір
Текстові	Алфавітно-цифрові дані.	0 – 255 символів
Поле МЕМО	Алфавітно-цифрові дані – речення, абзаци, тексти.	0 – 64 000 символів (байт)
Числові	Числові дані.	1, 2, 4 или 8 байт
Дата/ час	Дата та час.	8 байт
Грошовий	Дані про грошові суми, що зберігаються з 4 знаками після коми.	8 байт
Лічильник	Унікальне довге ціле, що генерується Access при створенні кожного нового запису.	4 байта
Логічний	Логічні дані: Так/Ні, Істина/Брехня.	1 бит (0 или -1)
Поле об'єкта OLE	Картинки, діаграми та інші об'єкти OLE із програм Windows.	До 1 Гбайта
Гіперпосилання	Адреса посилання (шлях) на документ або файл, що знаходиться на локальному комп'ютері, локальній мережі або мережі Інтернет.	До 2048 символів
Майстер підстановок	Інструмент для отримання значень з іншої таблиці або фіксованого списку значень.	—

Завдання загальних властивостей полів

Для кожного поля таблиці можна встановити значення властивостей, список яких залежить від вибраного типу даних. Наведемо найважливіші властивості полів.

Розмір поля (довжина поля) задає максимальний розмір даних, що зберігаються в полі.

Для поля з текстовим типом даних визначається розмір від 1 до 255 байт (за замовчуванням 50 байт).

Для поля з числовим типом даних можна встановити такі розміри:

– байт (1 байт) – для цілих чисел у діапазоні від 0 до 255;

- ціле (2 байти) - для цілих чисел в діапазоні від -32 768 до +32 767;

- Довге ціле (4 байти) - для цілих чисел в діапазоні від -2 147 483 648 до 2 147 483 647;

- з плаваючою точкою (4 байти) - для речових чисел від -3,4 · 1038 до +3,4 · 1038 з точністю до 6 знаків після коми;

- з плаваючою точкою (8 байт) - для дійсних чисел від -1,797 · 10308 до +1,797 · 10308 з точністю до 15 знаків після коми.

Рекомендується для значень поля задавати мінімально допустимий розмір, оскільки збереження таких полів вимагає менше пам'яті та обробка даних виконується швидше.

Необхідно запам'ятати, що зміна розміру з більшого на менший у таблиці, яка вже має дані, може призвести до їх спотворення або повної втрати.

Формат поля для певного типу даних визначає правила подання (відображення) їх при виведенні на екран або друк у режимі Таблиці, у формі або звіті. Access має вбудовані стандартні формати відображення для полів з типами даних: числовий, дата/час, логічний і грошовий. Деякі з цих форматів збігаються з настроюванням національних форматів, які визначаються у вікні Мова та регіональні стандарти панелі керування Microsoft Windows. За допомогою символів форматування користувач може створити власний формат для всіх типів даних, крім об'єкта OLE.

Число десяткових знаків для числового та грошового типів даних задає кількість десяткових розрядів після коми в діапазоні від 0 до 15.

Маска введення – дозволяє визначити власний формат, в якому слід вводити дані, та служить для полегшення введення та контролю форматованих даних. Маска є набір підстановочних знаків і текстових констант, який дозволяє управляти введенням даних. У табл. 2.2 наведені часто використовувані підстановочні символи (повний перелік підстановочних символів вказано у довідковій системі MS Access).

Таблиця 2.2

Символ	**Призначення**
0	У цю позицію слід ввести цифру.
9	У цій позиції може бути введена цифра або пробіл.
#	У цю позицію може бути введена цифра, пробіл, знак плюс (+) або мінус (–).
L	У цю позицію слід ввести літеру або цифру
&	У цю позицію слід ввести довільний символ або пробіл.
\	Вказує, що наступний за ним символ слід розглядати як постійний (виведений) символ, навіть якщо він є символом маски.

Підпис поля – задає текст, який виводиться у таблицях, формах, звітах.

Умова на значення - задає обмеження на значення, що вводяться, тим самим дозволяючи здійснювати контроль введення. У разі порушення заданої умови Access виводить повідомлення про причину припинення введення. Як умову можна встановити список допустимих значень.

Наприклад, вираз "Одеса" OR "Миколаїв" OR "Херсон" задає список допустимих назв міст. При введенні назви міста Access буде звіряти значення, що вводиться, із зазначеним в умові списком. Для завдання списку або інших складних умов рекомендується використовувати спеціальний інструмент, який є в Access, - Побудовник виразів.

Повідомлення про помилку дозволяє користувачеві сформулювати власний текст повідомлення, що відображається на екрані при порушенні обмежень, заданих властивістю Умова на значення.

Збереження таблиці

Створена структура таблиці має бути збережена. При збереженні необхідно встановити ім'я таблиці, після чого відбувається вихід з режиму Конструктора. Збережену таблицю тепер можна відкрити в режимі Таблиці та перейти до другого етапу створення таблиці введення даних (заповнення записів).

2.2.2 СТВОРЕННЯ ТАБЛИЦІ У РЕЖИМІ ТАБЛИЦІ.

Режим Таблиці дозволяє створити таблицю, не визначаючи попередньо її структури. Після вибору цього режиму відразу відкривається порожня таблиця, куди можна вводити дані. За збереження такої таблиці Access проаналізує дані та автоматично привласнить відповідний тип кожному полю, тобто створить структуру таблиці.

У режимі Таблиці можна перейменувати будь-яке поле таблиці відповідно до вимог користувача безпосередньо редагуючи імена в заголовку стовпців. Крім перейменування, допускається вставка, видалення чи зміна становища стовпців. Кожен стовпець вводяться дані певного типу. Для вводу типу даних необхідно використовувати формати зі встановленого списку стандартних форматів. Це дозволяє Access автоматично визначити тип даних та встановити значення властивості Формат поля за промовчанням.

2.2.3 СТВОРЕННЯ ТАБЛИЦІ ЗА ДОПОМОГОЮ МАЙСТРА ТАБЛИЦЬ.

Майстер таблиць автоматично створює таблицю за одним із шаблонів. Користувачеві пропонується більше 40 зразків таблиць, призначених для використання з різною метою. Кожна таблиця шаблону містить відповідний набір полів, у тому числі користувач може вибрати потрібні йому поля. Поля, що включаються в таблицю, можуть бути перейменовані.

Майстер таблиць визначить ключ таблиці, створить зв'язки нової таблиці з існуючими в базі даних. Після створення таблиці та її збереження можна будь-коли доопрацювати структуру таблиці як Конструктора.

2.3 ВИКОРИСТАННЯ РЕЖИМУ ТАБЛИЦІ ДЛЯ РОБОТИ З ДАНИМИ.

Таблиця БД, що відображається, відкрита в режимі Таблиці, складається з рядків (записів) і стовпців (полів). Імена полів розташовуються у верхній частині таблиці (у перших осередках кожного стовпця). Крайній ліворуч стовпець є полем виділення. Нижче заголовків стовпців (імен полів) слідують записи (рядки таблиці), у яких вносяться дані. Один запис завжди є поточним, ліворуч від неї в полі виділення розташований у вигляді стрілки покажчик поточного запису. Для створення нового запису служить останній рядок таблиці, зазначений у полі виділення зірочкою (*). У нижній частині вікна розташовані кнопки навігації, що дозволяють переміщувати покажчик поточного запису по таблиці (на перший запис, на попередній запис, наступний запис, останній запис). Там же знаходяться поле номера поточного запису, кнопка створення нового запису та вказівник загальної кількості записів у таблиці.

Студ3 : таблица

Номер	ФИО	Пол	ДатаРождения	ДомашнийАдрес	Факультет	ТелефонДеканата	ФИОдекан
1	Иванов И.Р.	М	12.03.1991	Ленинградская 113	ФПСВО	266-47-23	Черный Ю.Г.
2	Петрова Л.А.	Ж	30.01.1992	Лескова 112	ФПСВО	266-47-23	Черный Ю.Г.
3	Сидоров С.И.	М	05.08.1989	Тургенева 72	АСФ	266-43-37	Богатырева Т.Е
4	Королев М.П.	М	09.11.1988	Тургенева 16	ИЭФ	266-75-20	Гаршина Е.И.
5	Маркова Т.Е.	Ж	07.05.1987	Лескова 34	АСФ	266-43-37	Богатырева Т.Е
6	Лысов В.П.	М	15.12.1990	Тургенева 72	ФПСВО	266-47-23	Черный Ю.Г.
7	Прошин Т.Д.	М	31.01.1992	Московская 76	ФПСВО	266-47-23	Черный Ю.Г.
8	Островская М.	Ж	27.10.1992	Московская 32	ФПСВО	266-47-23	Черный Ю.Г.
9	Широкова Л.А.	Ж	16.11.1990	Толстого 64	ИЭФ	266-75-20	Гаршина Е.И.
10	Бобров А.Р.	М	30.09.1990	Кирова 56	АСФ	266-43-37	Богатырева Т.Е
11	Юров М.О.	М	23.11.1989	Московская 34	ИЭФ	266-75-20	Гаршина Е.И.
					ФПСВО		

Рис. 2.1. Відображення таблиці БД у режимі Таблиці

У Access існує безліч способів відображати лише один запис, групу записів або потрібні дані під час пошуку конкретного значення. Працюючи з одиночними таблицями застосовують такі операции:

Пошук та заміна даних. Для виконання цієї операції використовується спеціальний інструментальний засіб – діалогове вікно Пошук та заміна. Виконання операції пошуку дозволяє перейти до конкретного запису, що містить потрібне значення, і при необхідності замінити його новим значенням. Якщо використовувати діалогове вікно, пошук і заміна значень можуть здійснюватися по всій таблиці.

Сортування записів дозволяє змінити у таблиці порядок записів залежно від будь-якого критерію (поля). Якщо це поле має текстовий тип даних, то записи впорядковуються за абеткою, якщо числовий – у порядку зростання (зменшення) значень. Можливе сортування за кількома полями.

Фільтрування даних – застосування фільтрів, що дозволяють тимчасово ізолювати та переглянути конкретний набір записів відкритої таблиці, що задовольняють задану умову. При необхідності отриманий (відфільтрований) набір даних можна не лише переглянути, а й редагувати. Існують різні способи застосування фільтрів.

Фільтр по виділеному слід використовувати, якщо вдається легко знайти і виділити значення, яке повинні містити записи, що відбираються.

Фільтр за формою (звичайний фільтр) використовується для вибору значень зі списку без перегляду всіх записів, або при вказанні декількох умов відбору одночасно.

Фільтр для: використовується, якщо фокус (курсор) введення знаходиться в полі фільтра, в якому потрібно ввести конкретне значення або вираз, результат якого буде використовуватися як умова відбору. У полі цього фільтра можна вводити і маски, використовуючи символи (?) і (*).

Для створення складних фільтрів або сортування за декількома полями застосовується **Розширений фільтр**, який дозволяє здійснювати операції:

– пошуку записів, які задовольняють одночасно декільком умовам;

– пошуку записів, які задовольняють хоча б одному з кількох умов;

У разі використання розширеного фільтра відкривається спеціальне вікно (рис. 2.2). У верхній частині вікна (верхня панель) виводиться список полів таблиці, у нижній частині (нижня панель), що називається бланком, задаються для зазначених полів необхідні умови відбору та параметри сортування записів.

Зауваження. При виконанні сортування за декількома полями в бланку необхідно строго визначити порядок розташування полів, а саме, яке поле стоїть на першому місці, а які - на другому, третьому і т. д. місцях.

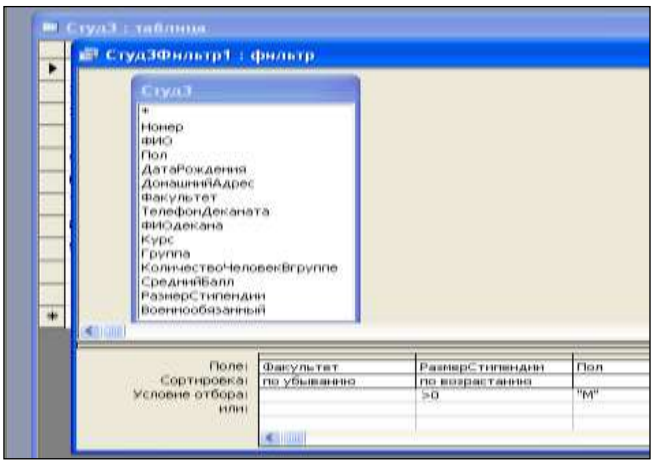


Рис. 2.2. Вікно розширеного фільтра.

3 ПОНЯТТЯ ЗАПИТУ СУБД. ОСНОВИ СТВОРЕННЯ ЗАПИТІВ.

Основне призначення таблиці в MS Access – зберігання внесених даних. У цьому вона грає роль пасивного сховища, т. е. сама неспроможна ініціювати процес запиту нових записів, ні передачу інформації кінцевому споживачеві. Для цього в СУБД служать інші об'єкти – запити.

Запити досить гнучкі і дозволяють розглядати дані так, як зручно для користувача.Запити можна використовувати:

- для перегляду підмножини записів таблиці без попереднього відкриття цієї таблиці чи форми;
- для того, щоб об'єднати у вигляді однієї таблиці дані з декількох таблиць (багатотабличний запит);
- перегляду окремих полів таблиці;
- до виконання обчислень над значеннями полів.

Запит виглядає як таблиця, хоч і не є нею. Записи такої таблиці є динамічний (результуючий) набір даних, який є віртуальним набором, тому що не зберігається в базі даних. Після закриття запиту результат набір даних припиняє своє існування. При збереженні запиту зберігається лише його структура – перелік таблиць, список полів, порядок сортування, обмеження запису, тип запиту. Щоразу, коли виконується запит, відбувається звернення до базових таблиць і створюється результуючий набір даних.

Оскільки сам собою результуючий набір даних не зберігається, запит автоматично відображає будь-які зміни, що відбулися в базових таблицях з моменту останнього запуску цього запиту. Принцип збереження структури запиту, а не результуючого набору даних має ряд переваг:

– на фізичному носії інформації (як правило, це жорсткий диск) потрібен менший обсяг простору;

– запит може використовувати оновлені версії будь-яких записів базової таблиці, змінених після останнього запуску запиту.

Усі запити поділяються на запити-вибірки та запити-дії.

Після виконання запиту-вибірки СУБД створює віртуальну таблицю, в яку заносить вибрану інформацію та зберігає її доти, доки згенерована таблиця не буде закрита. Коли цей запит закривається, отримана таблиця знищується, звільняючи пам'ять.

На відміну від запиту-вибірки, запит-дія змінює існуючі дані. За допомогою запиту-дії можна автоматично створити нову таблицю, внести дані до вже наявної таблиці, а також видалити або змінити набір записів з існуючої таблиці.

Будь-який новостворений запит у MS Access автоматично вважається запитом-вибіркою. У тому випадку, коли користувачеві або розробнику потрібно отримати запит, слід спеціально це вказати. Це допомагає уникнути випадкових дій із даними, які не можна скасувати.

Фактично, запит у MS Access є інструкцією на мові SQL (Structured Query Language), яка є на сьогоднішній день основним засобом складання запитів найбільш поширених реляційних СУБД. За допомогою механізму SQL стає можливим як звертатися до вмісту баз MS Access з інших додатків (зокрема і додатків з інших виробників), і отримувати дані для MS Access із зовнішніх додатків, підтримують SQL.

Запити в MS Access доступні трьох режимах: табличному, у якому запит виглядає як і, як звичайна таблиця, режимі конструктора, де запит постає як схеми пов'язаних об'єктів, і як інструкцій SQL.

Що спільногоі в чому різниця між фільтрами та запитами на вибірку? Подібність між фільтрами та запитами на вибіркуполягає в тому, що і в тих, і в інших інструментах проводиться вилучення підмножини записів з базової таблиці або іншого запиту. І запити на вибірку, і фільтри містять умови відбору даних таповертаютьвибірку, що відповідає зазначеним умовам, без зміни даних, що повертаються. Однак між ними існують відмінності, які потрібно розуміти, щоб правильно зробити вибір, у якому випадку використати запит, а в якому – фільтр. Основнівідмінностіфільтрів від запитівполягають у наступному.

– Фільтри працюють із відкритою таблицеюі зазвичай застосовуються в режимі Таблиці або в режимі Форми. Запити можуть використовуватись лише із закритою таблицею або іншим запитом.

– Фільтри не дозволяють в одному рядку відображати дані з кількох таблиць, тобто об'єднувати таблиці.

– Фільтрине дають змоги вказувати окремі поля, які мають відображатись у результуючому наборі записів. Вони завжди відображають усі поля базової таблиці.

– Фільтри не можуть бути збережені як окремі об'єкти БД. Запити є об'єктами БД і можуть зберігатися.

– Фільтри не дозволяють робити обчислення: підраховувати суми, середні значення, кількість записів та знаходити підсумкові значення.

3.1 ТИПИ ЗАПИТІВ.

У Access може бути створено кілька типів запиту.

Запит на вибірку –найпоширеніший тип запиту, у якому дані вибираються з однієї чи кількох взаємозалежних таблиць. Як джерело даних можна використовувати як таблиці БД, а й раніше створені запити, а вірніше, динамічний набір даних. Результати запиту оформляються у вигляді таблиці, записи якої формуються на основі умов відбору записів з вихідних таблиць і зв'язків між цими таблицями. Поля, які мають бути присутніми у запиті, вказуються користувачем. Ім'я запиту не повинно співпадати з ім'ям будь-якої таблиці БД, інакше створена запитом таблиця замінить існуючу. Таблиця з результатами запиту може застосовуватися для подальшої обробки даних. Наприклад, через таблицю запиту можна коригувати дані у вихідних таблицях БД.

Запити-дія*:*

Запит створення таблиці. За основу береться запит на вибірку, але, на відміну від нього, результат запиту зберігається в новій таблиці.

Запити на зміну– це запити на оновлення, додавання чи видалення. Вони складаються з інструкцій (операцій, команд), у результаті яких змінюються дані у вихідних таблицях. Запити на зміну дозволяють, виконавши одну операцію, внести зміни до багатьох записів вихідної таблиці.

Перехресний запит.Відображає результати статистичного аналізу даних, що зберігаються у таблиці. Вихідні дані групуються за двома наборами у форматі зведеної (перехресної) таблиці. Перший набір виводиться у стовпці зліва і утворює заголовки рядків, а другий виводиться у верхньому рядку та утворює заголовки стовпців.

Основні принципи створення запиту можна простежити техніці конструювання запиту вибірку, що є основою решти запитів.Access має кілька способів створення запитів: використання Конструктора запиту та Майстерів запитів.

Запит SQL— запит, який можна створити лише за допомогою SQL. Існує три типи таких запитів:

• запит-об'єднання — дозволяє об'єднати поля з кількох таблиць або запитів до одного набору даних;

• запит до сервера - передає інструкції SQL віддаленої бази даних;

• керуючий запит — Створює, змінює або видаляє таблиці або індекси бази даних Access.

3.2 СТВОРЕННЯ ЗАПИТУ.

3.2.1 МАЙСТРА СТВОРЕННЯ ЗАПИТУ.

Різні типи найпростіших запитів можуть бути створені за допомогою спеціальних майстрів запитів. Використання Майстра прискорює процес розробки запиту автоматично виконуючи початкові дії з підготовки цього запиту.

Запит створюється в інтерактивному режимі за кілька кроків роботи Майстра. На кожному кроці відображається діалогове вікно з інструкціями, які слід виконати користувачеві. В останньому вікні діалогу пропонується вибрати запуск або перегляд структури запиту в режимі Конструктора. За потреби можна відредагувати запит у режимі Конструктора.

За допомогою Майстра запитів можна створити:

- простий запит;
- запит для пошуку записів, що повторюються;
- запит для пошуку записів, які не мають підпорядкованих записів;
- *перехресний запит*.

Майстри запитів на вибірку.

Простий запит, а також запити для пошуку записів, що повторюються, і для пошуку записів, що не мають підлеглих, відносяться до типу елементарних запитів на вибірку.

*Простий запит.*Під час створення простого запиту Майстер надає можливість вибрати поля із взаємозалежних таблиць та запитів. У підготовленому бланку запиту не формуються умови відбору та поля, що обчислюються. Єдине, що може зробити Майстер – це підготувати підсумковий запит, згрупувавши записи та підрахувавши для цих груп суму, середнє значення, мінімум, максимум та кількість записів у групі.

*Запит для пошуку записів, що повторюються.*Майстерствоpення запиту для пошуку записів, що повторюються, будує запит, який визначає, чи містить таблиця повторювані значення в одному або декількох полях. Майстер дозволяє вибрати аналізовану таблицю, задати поля, в яких слід перевірити повторюваність значень, відібрати поля, які треба вивести разом з значеннями полів, що повторюються. Запит виводить ті записи, для яких є хоча б ще один запис у таблиці з однаковим значенням у вибраних полях.

Запит для пошуку записів, які не мають підлеглих.Майстер створення запиту для пошуку записів, що не мають підлеглих, дозволяє знайти в таблиці записи, у яких немає пов'язаних записів в підпорядкованій таблиці.

Майстер перехресних запитів

*Перехресний запит*створює таблицю даних у формі, подібної до електронної таблиці (зведеної таблиці). Але на відміну від інших запитів, що використовують як заголовки стовпців назви полів, перехресний запит як заголовки стовпців застосовує значення з таблиці. При роботі з перехресним запитом може виникнути потреба задати умови відбору записів для таких типів полів:

– для будь-якого нового поля;

– для поля заголовків рядків;

– для поля заголовків шпальт.

Перехресний запит будується нескладно як Конструктора. При цьому типі запиту в бланку запиту рядок Групова операція завжди активна, її не можна виключити. Цей рядок використовується для встановлення опції Угруповання, яка буде служити для вказівки заголовків рядків та стовпців.

Майстер перехресного запиту формує таблицю, в якій лівий стовпець утворює заголовки рядків із значень одного поля, верхній рядок утворює заголовки стовпців із значень іншого поля. На перетині рядків та стовпців розміщуються підсумкові значення, обчислені за значеннями третього поля. При цьому значення третього поля групуються за полями, що використовуються як заголовки, і для отриманих груп значень застосовується одна з вибраних статистичних функцій. З іншого боку, для рядків загалом може передбачатися виконання статистичних функцій.

3.2.2 СТВОРЕННЯ ЗАПИТУ У РЕЖИМІ КОНСТРУКТОРА. ЗАПИТ НА ВИБІРКУ.

За допомогою конструктора можна створити такі види запитів:

1. Простий.
2. За умовою.
3. Параметричні.
4. Підсумкові.
5. З полями, що обчислюються.

Щоб викликати Конструктор запитів, потрібно перейти у вікно бази даних. У вікні бази даних необхідно вибрати вкладку Запити та двічі клацнути на піктограмі Створення запиту в режимі конструктора. З'явиться активне вікно Додавання таблиці на фоні неактивного вікна "Запит: запит на вибірку".

У вікні Додавання таблиці слід вибрати таблицю - джерело або кілька таблиць з представленого списку таблиць, на основі яких проводитиметься вибір даних, і клацнути на кнопці Додати. Після цього закрити вікно Додати таблицю, вікно «Запит: запит на вибірку» стане активним.

Вікно Конструктора складається з двох частин – верхньої та нижньої. У верхній частині вікна розміщується схема даних запиту, що містить перелік таблиць – джерел і відбиває зв'язок з-поміж них.

У нижній частині вікна знаходиться Бланк побудови запиту, в якому кожен рядок виконує певну функцію:

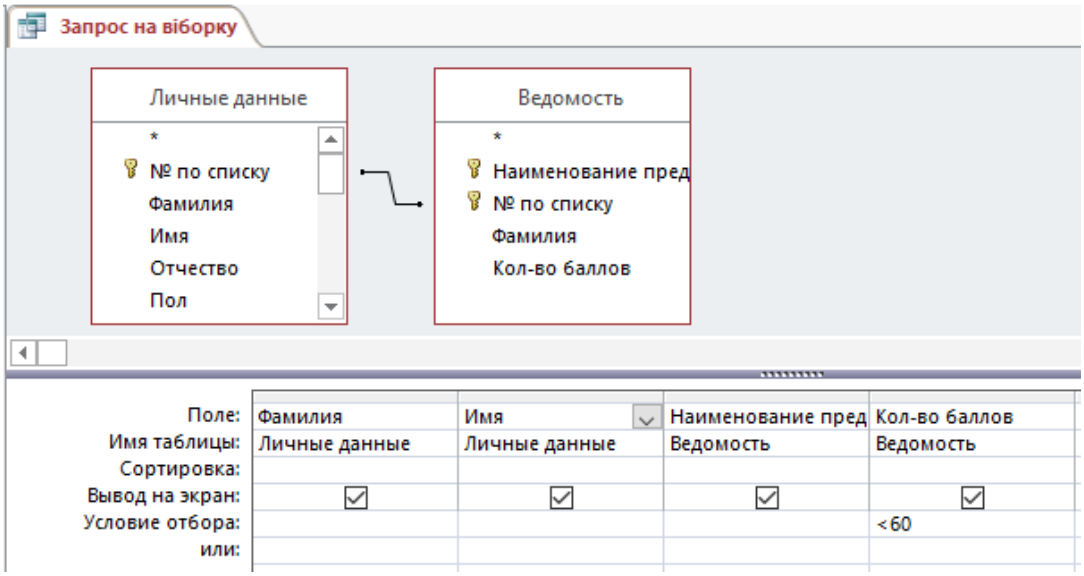
1. Поле – вказує імена полів, які беруть участь у запиті.
2. Ім'я таблиці – ім'я таблиці, з якої вибрано поле.

3. Сортування – вказує тип сортування.

4. Виведення на екран – встановлює прапорець перегляду поля на екрані.

5. Умови відбору – задаються критерії пошуку.

6. Або – задаються додаткові критерії відбору.



У вікні "Запит на вибірку" за допомогою інструментів формуємо запит:

1. Вибрати таблицю - джерело, з якої проводиться вибірка записів.

2. Перемістити імена полів з джерела в Бланк запиту.

3. Встановити принцип сортування. Курсор миші перемістити в рядок Сортування для будь-якого поля, з'явиться кнопка відкриття списку режимів сортування: за зростанням та за спаданням. У рядку виведення на екран автоматично встановлюється прапорець перегляду знайденої інформації у полі.

4. У рядку "Умови" відбору та рядку "Або" необхідно ввести умови обмеженого пошуку – критерії пошуку.*Умови відбору записів*– це обмеження, що накладаються на запит для визначення записів, з якими він працюватиме. Умови відбору можуть задаватися для одного або кількох полів. Щоб задати умову відбору для поля у відповідному рядку бланка, вводять вираз. *Вираз* включає операнди, з'єднані знаками математичних і логічних операцій, а також операцій порівняння. Як операнди можуть бути літерали, константи, змінні, функції, імена полів. В результаті обробки виразу можуть виконуватись обчислення, перетворення або перевірка даних. Введення виразів здійснюється або вручну, або за допомогою виразника.Accessвиконує синтаксичний аналіз введеного виразу та відображає його відповідно до результатів цього аналізу.

5. Після завершення формування запиту закрити вікно Запит на вибірку. Відкриється вікно діалогу Зберегти – відповісти Так (ввести ім'я створеного запиту, наприклад, Зразок запиту в режимі Конструктор) та клацнути ОК та повернутися у вікно бази даних.

У Access завжди можна додати нове поле в існуючий запит або видалити наявне поле. Для додавання поля до бланку запиту треба перетягнути його за допомогою миші з таблиці у потрібне місце бланка. При цьому відбувається зсув всіх стовпців, розташованих праворуч від місця вставки, на одну позицію праворуч. Для видалення непотрібного поля можна виділити стовпець і натиснути клавішу <Delete>, або використати відповідну команду.

3.3 ВИРАЗИ В ACCESS.

При роботі з різними об'єктами в Access широко використовуються вирази - аналог формул Excel. Вираз – це будь-яка комбінація операторів, констант, функцій та ідентифікаторів, результатом якої є певне значення. Константи, функції та ідентифікатори, що використовуються у виразах, називаються операндами.

Висловлювання найчастіше використовуються для перевірки різних умов та проведення обчислень у таблицях, запитах, формах та звітах. Вони дозволяють виконувати дії з числами, датами та текстовими значеннями кожного запису, використовуючи дані з одного або декількох полів. Наприклад, за допомогою виразу можна перемножити значення двох числових полів або об'єднати кілька текстових значень.

3.3.1 ОПЕРАТОРИ

У виразах застосовуються такі типи операторів:

•арифметичніоператори - використовуються для виконання математичних обчислень;

•оператори конкатенації- використовуються для злиття рядків;

•оператори порівняння- використовуються для виконання операцій порівняння;

•логічніоператори - використовуються для виконання логічних операцій;

• оператори ідентифікації -створюють однозначні імена об'єктів БД.

3.3.1.1 АРИФМЕТИЧНІ ОПЕРАТОРИ.

Операнди повинні бути виразами, що мають числове значення.

Для зміни пріоритету арифметичних операцій використовують круглі дужки. Якщо хоча б один з операнда є виразом зі значенням Null, то результат має значення Null.

Оператор поділу націло "\" округляє обидва операнди до цілих значень, а потім ділить перший на другий. Результат округляється до цілого,

наприклад, 11\2=5; 7,6\2,5 = 4.

Оператор Mod також округляє обидва операнди до цілих значень і ділить перший на другий. Результат – залишок від розподілу. Наприклад, 9 Mod 2 = 1, а 7,6 Mod 4 = 0.

Таблиця 3.1. Арифметичні оператори

Оператор	Опис	Приклад
+	Складає два операнди	[ціна] + 10
-	Віднімає з першого операнда другий	[Дата1] – [Дата2]
	або змінює знак операнда	-111
*	Перемножує два операнди	[Ціна]*[Вага]
/	Ділить один операнд на другий	[Сума] / 10
\	Делить один операнд на другий націло	[місяць] \4
^	Зводить перший операнд у ступінь,задається другим операндом	[Число] ^ [Ступінь]
Mod	Повертає залишок від поділу націло	[Місяць] mod 4

3.3.1.2 ОПЕРАТОРИ ЗЛИТТЯ РЯДКІВ (КОНКАТЕНАЦІЇ).

Оператори & (амперсанд) або+ створюють текстовий рядок, приєднуючи вміст другого рядка до кінця першого. Якщо один з операнда - число, то він перетворюється перед проведенням операції злиття в рядок символів.

Для об'єднання рядків краще використовувати оператор &, а не +, тому що якщо один з рядків, що беруть участь в операції, наприклад <рядок 2>, має значення Null, то результат операції <рядок 1> + <рядок 2> дорівнює Null, , а результат операції <рядок1> & <рядок 2> дорівнює <рядок 1>.

Наприклад, у виразі " Кількість замовлень = " & [Кількість замовлень] об'єднують рядок символів і значення поля Кількість замовлень. Якщо кількість замовлень дорівнює 100, то результатом виконання операції буде рядок

"Кількість замовлень = 100".

3.3.1.3 ОПЕРАТОРИ ПОРІВНЯННЯ.

Оператор порівняння порівнює значення двох операндів і повертає як результат одне з логічних значень: True або False. Якщо хоча б один з операнда є виразом зі значеннямNull, то результат має значенняNull.

Таблиця 3.2. Оператори порівняння

Оператор	Опис	Приклад	Результат
<	Менше ніж	1+2< 3+4	True
<=	Менше або дорівнює	1 <= 3/5	False
>	Більше ніж	1> 0	True
>=	Більше або дорівнює	2 >= 1	False
=	Рівно	1= 1	True
<>	Не дорівнює	3 <> 1	False

Крім стандартних операторів порівняння, наведених у таблиці 3.2, в Access є ще чотири додаткові оператори порівняння: Is, In, Between та Like, які зазвичай використовуються для перевірки умови на значення в полі або в умовах відбору записів у запиті.

Оператор Is

При використанні разом з Null визначає, чи значення Null або Not Null. Наприклад, Is Null застосовується для відбору записів, що мають у даному полі значення Null, а Is Not Null – для відбору записів, що мають у даному полі непусте значення .

Оператор In

Здійснює перевірку значення на збіг з елементом із заданого списку. Елементи списку відокремлюються один від одного крапкою з комою. Текстові значення повинні братися в лапки, наприклад,

In("Одеса";"Київ";"Херсон") або In(2;4;6;8).

Оператор Between

Здійснює перевірку, чи є числове значення всередині заданого діапазону. Наприклад, Between 10 And 20 означає, що значення має бути в інтервалі [10, 20].

Оператор Like

Здійснює перевірку значення відповідності заданому шаблону.

Таблиця 3.3. Спецсимволи, що використовуються в операторі Like

Спецсимвол	Збігаючі символи
?	Будь-який одиночний символ;
#	Будь-яка одиночна цифра (0-9);
*	Будь-яке число символів чи їх відсутність;
[перелік]	Будь-який одиночний символ, що входить до списку;
[!перелік]	Будь-який одиночний символ, який не входить до списку.

Таблиця 3.4. Приклади використання оператора Like

Умова	Коментар
Like "А*ів"	Будь-який текст, що починається з літери «А» тазакінчується літерами «ів»;
Like "К??#"	Значення має містити чотири символи; починатися з літери К та закінчуватися цифрою.
Like "[А-ВК]*"	Будь-який текст, що починається з літер А, Б, В та К;
Like "[!П-СЯ]*"	Будь-який текст, що не починається з літер П, Р, С та Я.

У таблиці 3.3 перераховані спеціальні символи, що використовуються в шаблоні, і відповідні символи в порівнюваному виразі. При перевірці збігу символів їхній регістр ролі не грає. Щоб включити до списку діапазон символів, потрібно вказати перший символ, знак дефісу, а потім останній символ, наприклад [К-Р].

3.3.1.4 ЛОГІЧНІ ОПЕРАТОРИ

Зазвичай застосовуються для об'єднання двох або кількох умов у єдине ціле. Нижче наведені найчастіше використовувані логічні оператори.

Таблиця 3.5. Логічні оператори

Оператор	Призначення
And	Повинні виконуватись всі умови;
Or	Повинно виконуватись хоча б одна з умов;
Not	Не повинна виконуватись ця умова.
Eqv	Повертає значення «істина», якщо Вираз1 і Вираз2 істинні або Вираз1 і Вираз2 помилкові.

3.3.1.5 ОПЕРАТОРИ ІДЕНТИФІКАЦІЇ.

Часто у висловлюваннях використовуються значення полів таблиць, елементів управління форм та інших об'єктів БД. Імена полів у різних таблицях або елементів

керування у формах можуть збігатися. Щоб Access правильно обчислив значення висловлювання, необхідно забезпечити однозначність посилань на об'єкти БД та його властивості.

Access використовує два оператори ідентифікації "!" (знак оклику) і "." (крапка).

Оператор "!"

Найчастіше в ідентифікаторах зустрічається оператор "!" Він використається для посилань на об'єкти. При посиланні на полі таблиці він служить відділення імені поля від імені таблиці. Самі імена полягають у квадратних дужках, і посилання має такий вигляд:

[<ім'я таблиці>]![<ім'я поля>].

Це так звана повна форма запису ідентифікатора поля таблиці. Якщо немає невизначеності у посиланні, то допустима і неповна форма запису ідентифікатора поля таблиці як [<ім'я поля>]. Так, у запитах, які використовують одну таблицю, зазвичай використовується неповне посилання на полі. Наприклад, повне посилання на полеПрізвище в таблиці

Студент має вигляд:

[Студент]![Прізвище], а неповне - [Прізвище].

Відповідно, посилання на елемент управління форми (головної форми, якщо вона містить підлеглу форму) має таку повну форму запису:

Forms![<ім'я форми>]![<ім'я елемента керування>].

Тут імені форми передує ім'я сімейства відкритих форм Forms, якому належить ця форма. Ця «добавка» викликана тим, що БД цілком може містити таблицю і форму з однаковими іменами, що мають до того ж однойменні поля.

У випадку синтаксис оператора «!» такий:

<клас об'єкта>!<ім'я об'єкта>

Оператор "."

Оператор "." (Точка) зазвичай використовується для посилань на властивості об'єкта (форм, звітів та елементів управління). Зокрема, ідентифікатор поля зі списком у формі має такий вигляд:

Forms![<ім'я форми>]![<ім'я поля зі списком>].[Text].

Тут точка "." використовується для відокремлення імені поля зі списком від властивості Text, яке повертає поточне значення поля. У загальному випадку синтаксис оператора "." такий: <клас об'єкта>!<ім'я об'єкта>.<властивість об'єкта>

3.3.2. КОНСТАНТИ

Увиразах зустрічаються константи наступних типів:

• Числові константи (числа) - послідовність цифр, що містить, якщо потрібно, знак числа і роздільник цілої та дробової частини числа. Як роздільник в залежності від установок Windows зазвичай використовуються "," (кома) або "." (Десяткова точка). Числа можуть містити символ і знак порядку, наприклад,

1,2 E +04 = 12000.

• Текстовіконстанти (рядки) - можуть містити будь-які символи набору символів кодової таблиці ANSI. У виразах рядка потрібно з обох боків укладати в прямі лапки ("). Зазвичай Access додає їх сам.

• Константи типу Дата/час.Вони мають бути поміщені у знаки номера "#". Зазвичай Access додає їх сам, якщо впізнає, що вводиться дата в одному із стандартних форматів "дд.мм.гг" або "дд/мм/гг".

3.3.3 ФУНКЦІЇ

До складу виразів часто входять різні функції. Усього в Access і VBA визначено понад 160 функцій. Тут наводиться короткий опис лише частини їх.

Функції для роботи з датами.

•Date()- Поточна дата. Може використовуватись у формах та звітах, а також задавати умову відбору для запиту.

Наприклад,Date() – 1 визначає дату, що передує поточній даті.

• Day(дата 1)— день місяця, ціле число від 1 до 31.

Наприклад, Day(#10.08.99#) повертає номер дня, що дорівнює 10.

• Month(дата)— місяць, ціле число від 1 до 12.

Наприклад, Month(#10.08.99#) повертає номер місяця, що дорівнює 8 (серпень).

• Now()- Дата та час комп'ютера. Часто використовується у звітах, створених за допомогою майстрів Access.

• Weekday(дата)день тижня, ціле число від 1 до 7, неділя дорівнює 1.

Наприклад, Weekday (# 22.03.73 #) повертає число 5 (четвер).

• Year(дата) - рік, ціле число.

Наприклад,Year([Студенти]![Дата народження])повертає рік народження студента.

Функції для роботи з рядками.

• Chr(код_символу)— повертає символ, який відповідає коду символу з кодової таблиці Windows ANSI.

Наприклад, Chr (100) = "d", Chr (200) = "I".

• Left (рядок, число_символів)- Вказана кількість перших символів рядка.

Наприклад, Left ([Студенти]! [Ім'я], 1) повертає першу букву імені студента.

• Len (рядок) -число символів у рядку.

Наприклад, Len([Прізвище]) дає число символів у прізвищі, що міститься в полі Прізвище.

• Mid(рядок; поч_символ; число_символів) — повертає підрядок,містить вказану кількість символів рядка, починаючи з вказаного символу. Останній аргумент необов'язковий. Якщо він відсутній, то повертаються всі символи, починаючи з зазначеного символу до кінця рядка.

Наприклад, Mid ("Студент Петров"; 9; 4) повертає "Петро", а Mid ("Студент Петров"; 9) повертає "Петрів".

• Right(рядок, число_символів) — кількість останніх символів рядка. Наприклад,Right([Код студента],3) повертає останні три символи коду студента.

• Trim(рядок)— видаляє пробіли на початку та в кінці рядка символів.

Математичні функції.

• Abs (вираз) -повертає абсолютне значення числового аргументу.

Наприклад,Abs(-10) = 10.

• Int (вираз) -повертає цілу частину числового аргументу <вираз>.

Наприклад,Int(5,2) = 5

• Rnd() -повертає випадкове число між 0 та 1.

Статистичні функції.

Повертають як значення результат відповідної статистичної операції над даними, що містяться у вказаному полі запиту, форми або звіту. Записи з порожніми (Null) значеннями у полі обчисленнях не беруть участь. Зазвичай використовуються у підсумкових запитах, при створенні обчислюваних полів та інструкції SQL.

• Avg (вираз 1) -обчислює середнє арифметичне значень, які у зазначеному полі. Наприклад, Avg([Стипендія]! [Вересень]) знаходить середню стипендію у вересні.

• Count (вираз) -визначає кількість записів, які повертаються запитом. Наприклад,Count([Стипендія]![Жовтень]) обчислює кількість студентів, які отримали стипендію у жовтні.

• First (вираз) -повертає значення, яке міститься у вказаному полі першого запису результату запиту. Зазвичай результат запиту попередньо сортується. Наприклад,First([Студенти]![Прізвище]) знаходить прізвище наймолодшого студента, якщо записи в запиті відсортовані за спаданням у полі [Дата народження].

• Last(вираз) — повертає значення, яке міститься у вказаному полі останнього запису результату запиту. Наприклад, Last([Студенти]![Прізвище]) знаходить прізвище наймолодшого студента, якщо записи у запиті відсортовані за зростанням у поліДата народження.

• Max (вираз) -підраховує максимальне значення, що містяться у зазначеному полі. Наприклад,Max([Студенти]![Дата народження]) знаходить дату народження наймолодшого студента.

• Min (вираз) -підраховує мінімальне з набору значень, які у зазначеному полі. Наприклад,Min([Книги]![Цена]) знаходить мінімальну цін на книги.

• StDev (вираз)- Повертає значення незміщеної оцінки стандартного відхилення значень, що містяться у вказаному полі.

• Sum (вираз)— підраховує суму значень, які у зазначеному полі. Наприклад, Sum ([Замовлення]! [Кількість] * [Книги]! [Ціна]) обчислює сумарну вартість замовлень.

• Var (вираз)— повертає значення незміщеної оцінки дисперсії значень, які у зазначеному полі.

Інші корисні функції.

• Is Null (вираз) - повертаєTrue, якщо <вираз> має значення Null; в іншому випадку функція повертає значення False. Наприклад, значення IsNull ([Стипендія]! випадку.

• IIF(умова; вираз1; вираз2) - повертає значення<вираз1>, якщо умова дорівнює Truei <вираз2>, якщо умова дорівнює False. Наприклад, IIF([Пол]="м"; "студент"; "студентка")

має значення «студент», якщо в поліПол міститься буква "м", і значення «студентка» - інакше.

• Nz(вираз[; подання]) — повертає 0 (нуль), порожній рядок ("") або інше вказане в аргументі<подання> значення, якщо <вираз> має значення Null.

Наприклад, Nz(([Стипендія]! [Вересень];"ні")повертає значення стипендії студента за вересень, якщо він у вересні отримував стипендію, або слово "ні" в іншому випадку.

Аргумент <подання> необов'язковий. Якщо він відсутній, то функція Nz повертає нуль або порожній рядок залежно від контексту, що потребує числового чи текстового значення.

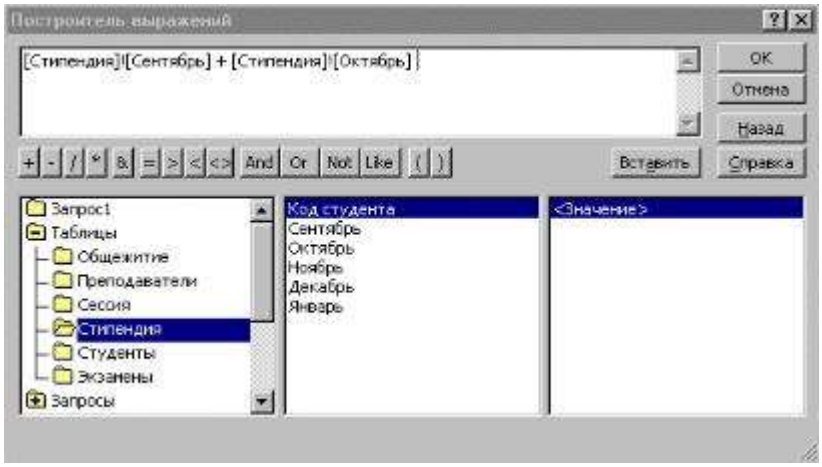
3.4 ПОБУДОВНИК ВИРАЗІВ.

Під час створення виразів для таблиць, запитів та інших об'єктів Access слід використовувати будівельник виразів. Для виклику будівельника потрібно спочатку клацнути по осередку, в який вводитиметьсявираз, а потім за кнопкою Побудувати на панелі інструментів або на кнопці зазвичай з'являється праворуч від осередку введення. На екрані з'явиться вікно Побудовник виразів, що містить чотири поля.

У верхньому полі розташовується вираз, а три нижніх використовуються для вибору потрібних елементів. Для вибору будь-якого елемента в цих полях потрібно зробити подвійне клацання по відповідному імені.

Ліве поле відображає ієрархію папок, що містять основні типи виразів. Після вибору елемента (папки) з лівого поля в середньому поле буде виведено або список елементів (поля таблиці або запиту), або список підтипів (елементи управління форми, категорії функцій тощо). При виборі підтипу в правому полі з'явиться список елементів (поля, функції, властивості елементів управління).

Щоб ввести посилання, що формується, на ім'я поля таблиці або запиту, потрібно вибрати в лівому полі таблицю або запит, а потім в середньому полі потрібне поле.



Щоб ввести функцію, виберіть у лівому полі папку Опції та Вбудовані функції. У середньому полі потрібно вибрати категорію або варіант <Все>, а потім, прокрутивши список у правому полі, потрібну функцію.

Для введення оператора (+, >, And та ін.) клацніть на відповідній кнопці у вікні будівельника. Якщо потрібного оператора на кнопках немає, відкрийте папку Оператори в лівому полі. Потім у середньому полі вибрати категорію або варіант <Все>, а правому полі — потрібний оператор.

Access часто вставляє у вираз, що створюється, разом з вибраним елементом один або кілька прототипів, укладених у лапки («вираз», «number» тощо). У цьому випадку необхідно або ввести замість прототипу відповідне значення, або виділити прототип і замінити його елементом з правого списку, або видалити його.

Для вставлення елемента у вираз можна використати кнопку Вставити. Щоб скасувати помилкове введення, натисніть кнопку Назад. Створення виразу завершується натисканням кнопки OK.

3.5 ПАРАМЕТРИЧНІ ЗАПИТИ.

Конкретне значення поля за умови відбору може вводитися безпосередньо до бланку запиту або задаватися користувачем у спеціальному діалоговому вікні під час виконання запиту. Щоб виводити діалогове вікно для введення конкретного значення поля, потрібно визначити параметр запиту.

Ім'я параметра запиту може вводитись у квадратних дужках у бланк запиту безпосередньо в рядок Умова відбору. При виконанні запиту це ім'я з'явиться у діалоговому вікні Введіть значення параметра.

Якщо запит вводиться кілька параметрів, то порядок їх введення через діалогові вікна визначається порядком розташування полів з параметрами в бланку запиту.

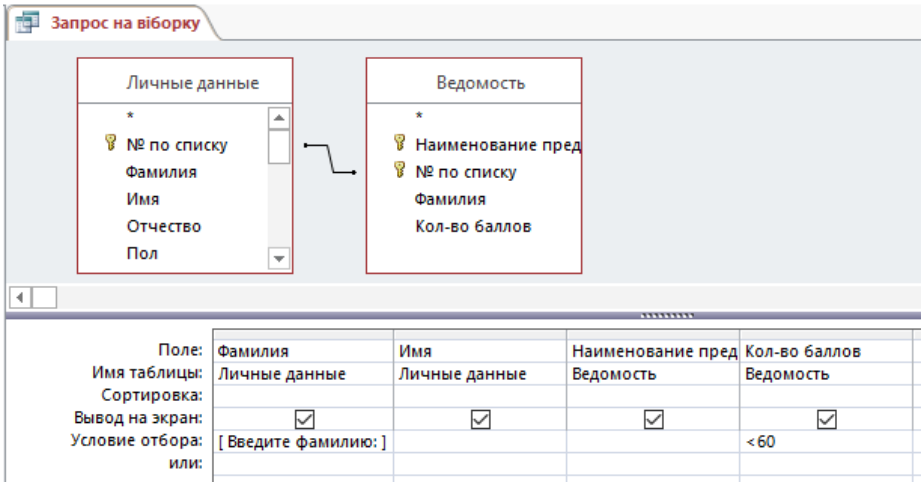
Для того, щоб мати можливість при виконанні запиту ввести кілька значень для одного поля, можна за умови відбору цього поля визначити кілька параметрів, зв'язавши їх логічними операціями. Наприклад, для відбору записів за двома групами за умови відбору можна записати два параметри, пов'язані логічною операцією OR.

Запити, що являють собою варіанти базового запиту і незначні один від одного, називаються параметричними. У параметричному запиті вказується критерій, який може змінюватись на замовлення користувача.

Послідовність створення параметричного запиту:

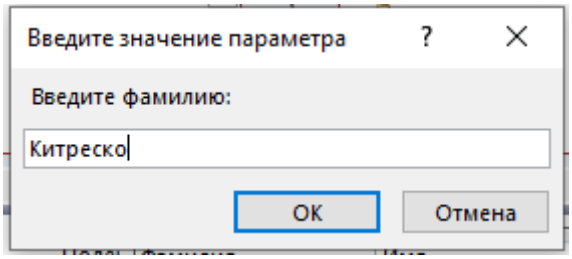
1. Створити запит у режимі конструктора або відкрити існуючий запит у режимі конструктора.

2. У Бланк запиту в рядку Умови відбору ввести умову відбору як запрошення у квадратних дужках, наприклад: [Введіть прізвище].



3. Закрити вікно Запит на вибірку, питання збереження зміни відповісти – Так. Повернутись у вікно бази даних, де створений запит буде виділено.

4. Виконати запит, натиснувши. У вікні діалогу «Введіть значення параметра», що з'явиться на екрані, треба ввести, наприклад прізвище студента, інформацію про успішність якого необхідно отримати, виконати клацання по кнопці ОК.



Результат виконання запиту:

Запрос на вибірку

Фамилия	Имя	Наименование предмета	Кол-во баллов
Китреско	Никита	Автоматизація виробничих процесів	50
Китреско	Никита	Моделювання процесів і систем	30

3.6 ОБЧИСЛЮВАЛЬНІ ПОЛЯ.

У запиті над полями можуть проводитись обчислення. Результат обчислення утворює поле, що обчислюється, в динамічному наборі даних цього запиту.

Поле, вміст якого є результатом розрахунку за вмістом інших полів, називається полем, що обчислюється. Загальний формат поля, що обчислюється, виглядає так:

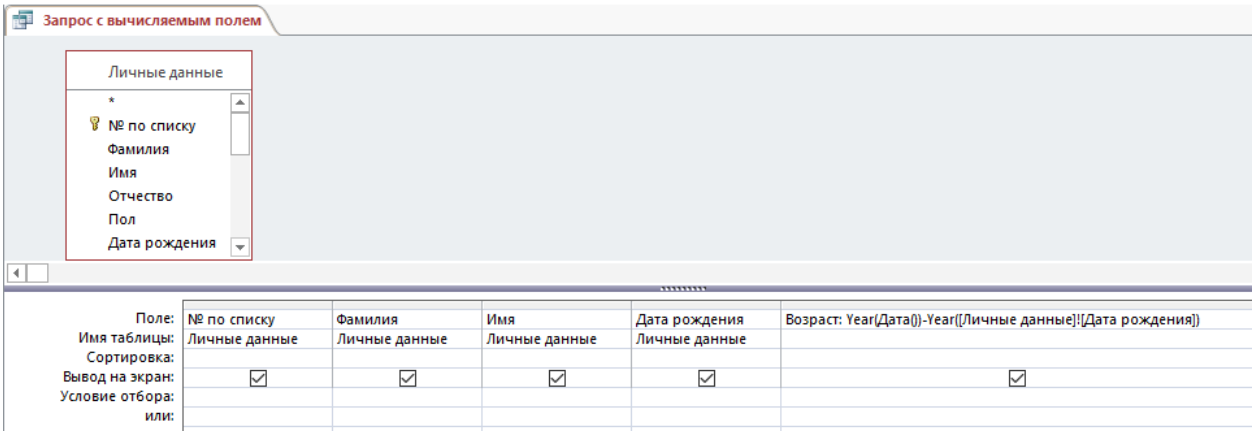
Ім'я поля, що обчислюється: Вираз для створення обчислюваного поля.

Користувач має можливість задавати власні імена полям, що обчислюються. При кожному виконанні запиту провадиться обчислення з використанням поточних значень полів із таблиці БД.

У полях, що обчислюються, можна використовувати вбудовані функції, які застосовуються до полів підмножини записів. Наприклад, функція Date формує поточну дату; функція Dlookup повертає значення конкретного поля із запису пов'язаної таблиці, що не бере участь у запиті. Широко використовуються статистичні функції, що обчислюють середнє значення, суму, мінімальне та максимальне значення.

Для полів, що обчислюються, як і для будь-яких інших полів, допускається сортування, завдання умов відбору та розрахунок підсумкових значень.

Приклад: Визначити вік студентів.



Результат виконання запиту:

Запрос с вычисляемым полем

№ по списк	Фамилия	Имя	Дата рожден	Возраст
1	Бессараб	Владимир	01.01.1996	26
2	Едалов	Дмитрий	26.07.1993	29
3	Каратаев	Вячеслав	03.04.1995	27
4	Китреско	Никита	14.05.1997	25
5	Нитченко	Назар	23.09.1996	26
6	Сурняк	Роберт	01.03.1996	26
7	Тельпис	Роман	07.07.1997	25
8	Тихонович	Денис	07.08.1996	26
9	Тарасов	Дмитро	06.05.1995	27
10	Халецька	Альона	24.01.1995	27
11	Саввіч	Олексій	04.01.1996	26
12	Шувалова	Елена	01.06.1996	26
13	Яворская	Ольга	01.01.1995	27
14	Некрасов	Николай	01.01.1996	26

3.7 ВИКОРИСТАННЯ ГРУПОВИХ ОПЕРАЦІЙ У ЗАПИТАХ.

Часто необхідно знайти інформацію, що базується на узагальнених даних. Наприклад, потрібно дізнатися загальну кількість виконаних замовлень або загальну кількість вантажів, надісланих на замовлення минулого року. Access надає повний набір інструментів для складання таких запитів.

Такі запити є підсумковими і реалізуються завдяки використанню статистичних функцій, що визначають потрібні величини вмісту поля. Статистичні розрахунки можна виконувати як з усіх записами, і над групами записів (групові операції) з однієї чи кількох таблиць. Результат запиту з використанням групових операцій містить один запис для кожної групи. До запиту включаються поля, за якими виконується групування, та поля, для яких виконуються групові функції.

Щоб виконати групові обчислення для кожного включеного в запит поля, в режимі Конструктора в бланку запиту слід вибрати одну з опцій у списку, що розкривається, що знаходиться в рядку Групова операція:. Дванадцять опцій цього списку можна розділити на чотири категорії, представлені в табл. 3.6.

Таблиця 3.6

Категорія	**Кількість опцій**	**Призначення операції**
Угруповання	1	Збирає записи, що мають загальні ознаки, групи, над якими згодом проводяться обчислення. Згруповані записи використовуються в запиті для виконання сумісних обчислень над іншими записами.
Набір операцій	9	Позначають математичну операцію, яка буде виконана над полем.
Вираз	1	Об'єднує кілька операцій у вираз. Поле обробляється у кілька етапів.
Умова	1	Накладає будь-яке обмеження. Встановлюються обмеження критерії для полів, над якими будуть виконані статистичні розрахунки.

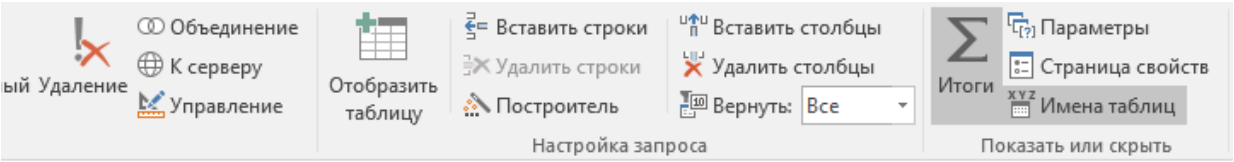
Категорія Набір операцій дозволяє виділити групи записів з однаковими значеннями у зазначених полях та використовувати для інших полів цих груп певну статистичну функцію. У Access передбачається дев'ять статистичних функцій:

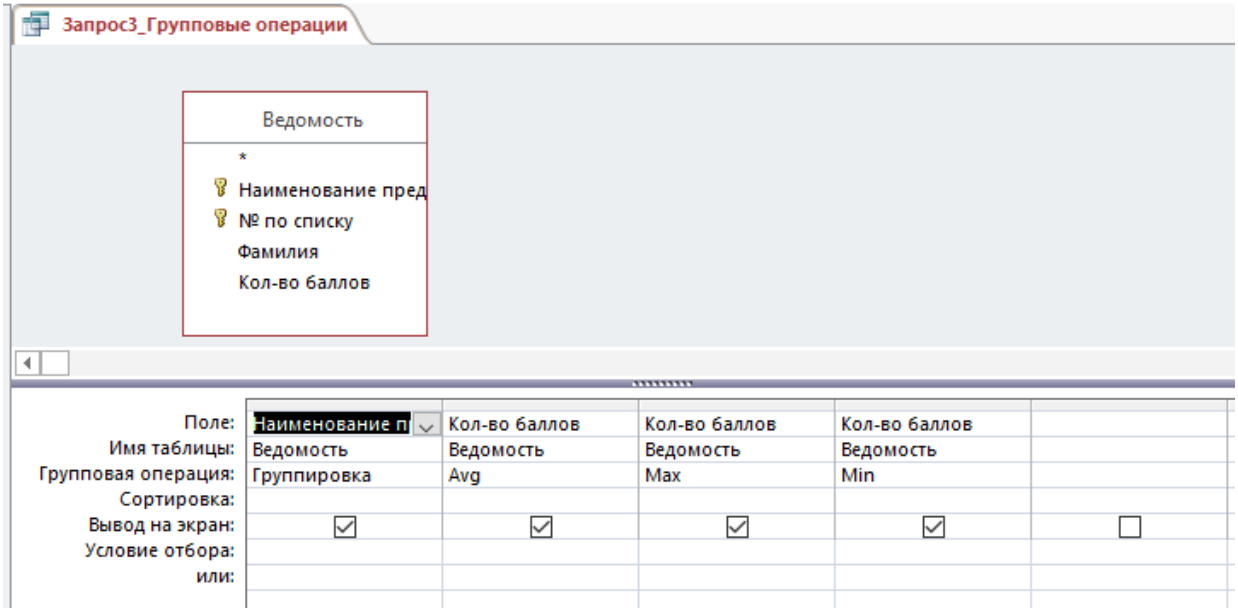
- *Sum* сума значень деякого поля для групи;
- *Avg* середнє від усіх значень поля у групі;
- *Max, Min* максимальне або мінімальне значення поля у групі відповідно;
- *Count* число значень поля групи без урахування порожніх значень;
- *StDev* середньоквадратичне відхилення від середнього значення поля у групі;
- *Var* дисперсія значень поля у групі;
- *First, Last* значення поля з першого або останнього запису групи.

Приклад : Визначити середній, максимальний, мінімальний бали здачі сесії з кожного предмета.

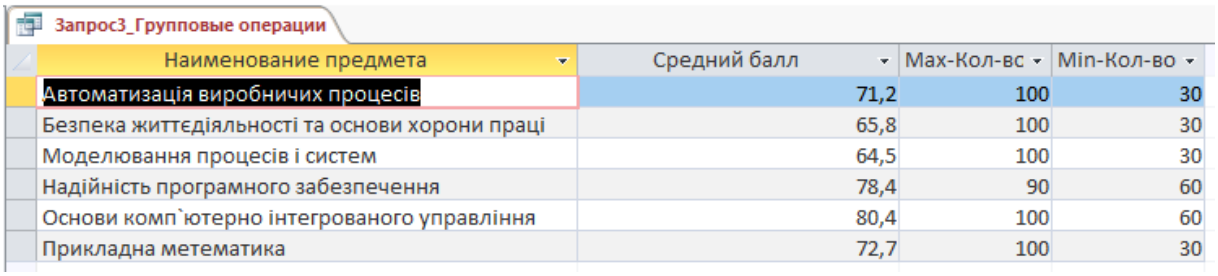
Для цього:

1. Натисніть кнопку підсумки. У бланку запиту буде додано поле Групові операції.
2. Виберіть потрібні функції для вирішення завдання.





Результат виконання запиту:



Наименование предмета	Средний балл	Max-Кол-вс	Min-Кол-во
Автоматизація виробничих процесів	71,2	100	30
Безпека життєдіяльності та основи хорони праці	65,8	100	30
Моделювання процесів і систем	64,5	100	30
Надійність програмного забезпечення	78,4	90	60
Основи комп`ютерно інтегрованого управління	80,4	100	60
Прикладна метематика	72,7	100	30

3.8 РОБОТА З СТАТИСТИЧНИМИ ФУНКЦІЯМИ З ПІДМНОЖНОСТІ.

Статистичні функції за підмножиною є аналогами підсумкових функцій, що входять до групових операцій. Але на відміну від підсумкових функцій ці функції можуть містити умови відбору оброблюваних ними записів. Як умову відбору використовуються такі ж логічні висловлювання, що й у запитах. ACCESS має такі статистичні функції:

DAvg- Підрахунок середнього арифметичного значення стовпця або виразу,

DCount- Підрахунок кількості записів,

DFirst- Знаходження першого значення стовпця з групи,

DLast- Знаходження останнього значення стовпця з групи,

DMax- Визначення максимального значення стовпця або виразу,

DMin- Визначення мінімального значення стовпця або виразу,

DSum- Підрахунок суми значень стовпця або виразу.

Синтаксис операторів є наступним.

<ім'я_функції> ("аргумент1"; "аргумент2";"аргумент3")

Функції мають два обов'язкові аргументи та один необов'язковий. Усі аргументи текстового типу.

Аргумент1- є обов'язковим та визначає ім'я стовпця або вираз, над значеннями якого виконуватиметься функція. Цей аргумент має бути рядком символів, тобто повинен полягати у лапки.

Аргумент2– також є обов'язковим та визначає ім'я таблиці або запиту (джерело записів), з яких Вибираються стовпці або обчислюються вирази.

Аргумент3– «Критерій відбору» є необов'язковим. Якщо він присутній, то

визначає, яких записів Обчислюється функція, якщо відсутня, то функція обчислюється за всіма записами.

Приклади використання функцій: Вихідними даними є відомість здачі сесії групи.

Ведомость

Наименование предмета	№ по списк	Фамилия	Кол-во баллов
Автоматизація виробничих процесів	1	Дмів	75
Автоматизація виробничих процесів	2	Жуковський	90
Автоматизація виробничих процесів	3	Ільясов	70
Автоматизація виробничих процесів	4	Кваша	10
Автоматизація виробничих процесів	5	Маловичко	100
Автоматизація виробничих процесів	6	Петлінський	70
Автоматизація виробничих процесів	7	Радалов	60
Автоматизація виробничих процесів	8	Серт	60
Автоматизація виробничих процесів	9	Тарасов	100
Автоматизація виробничих процесів	10	Халецина	90
Автоматизація виробничих процесів	11	Саввін	10
Автоматизація виробничих процесів	12	Шувалов	80
Автоматизація виробничих процесів	13	Яворський	70
Безпека життєдіяльності та основи х	1	Дмів	90
Безпека життєдіяльності та основи х	2	Жуковський	65
Безпека життєдіяльності та основи х	3	Ільясов	70
Безпека життєдіяльності та основи х	4	Кваша	60
Безпека життєдіяльності та основи х	5	Маловичко	30
Безпека життєдіяльності та основи х	6	Петлінський	70
Безпека життєдіяльності та основи х	7	Радалов	90
Безпека життєдіяльності та основи х	8	Серт	60
Безпека життєдіяльності та основи х	9	Тарасов	100
Безпека життєдіяльності та основи х	10	Халецина	60
Безпека життєдіяльності та основи х	11	Саввін	20
Безпека життєдіяльності та основи х	12	Шувалов	80
Безпека життєдіяльності та основи х	13	Яворський	70
Моделювання процесів і систем	1	Дмів	80
Моделювання процесів і систем	2	Жуковський	80
Моделювання процесів і систем	3	Ільясов	80
Моделювання процесів і систем	4	Кваша	30
Моделювання процесів і систем	5	Маловичко	80
Моделювання процесів і систем	6	Петлінський	70
Моделювання процесів і систем	7	Радалов	70
Моделювання процесів і систем	8	Серт	70
Моделювання процесів і систем	9	Тарасов	100
Моделювання процесів і систем	10	Халецина	50
Моделювання процесів і систем	11	Саввін	30
Моделювання процесів і систем	12	Шувалов	60
Моделювання процесів і систем	13	Яворський	70

Приклад 1:

Визначити мінімальний бал, одержаний студентами за сесію. Вивести список предметів та студентів, які мають мінімальну оцінку.

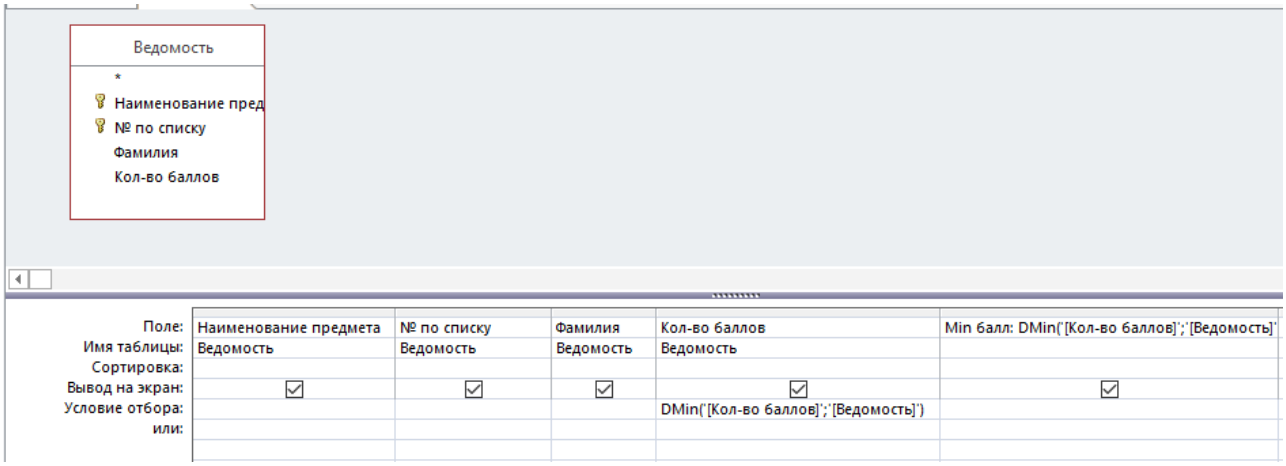
Для цього:

1. Визначаємо мінімальний бал по сесії та виводимо його в окремому полі:

Min бал: DMin('[Кількість балів]';'[Відомість]')

2. Як умову відбору для поля «Кількість балів» додаємо вираз:

DMin('[Кількість балів]';'[Відомість]')



Результати виконання запиту:

Наименование предмета	№ по списк	Фамилия	Кол-во баллов	Min балл
Автоматизація виробничих процесів	11	Саввіч	30	30
Безпека життєдіяльності та основи хорони праці	5	Маловичко	30	30
Безпека життєдіяльності та основи хорони праці	11	Саввіч	30	30
Моделювання процесів і систем	4	Кваша	30	30
Моделювання процесів і систем	11	Саввіч	30	30
Прикладна метематика	2	Жуковський	30	30
Прикладна метематика	7	Радалов	30	30
*	0		0	

Приклад 2:

Необхідно визначити студентів, середній бал яких вищий за середній бал по групі.

Для цього:

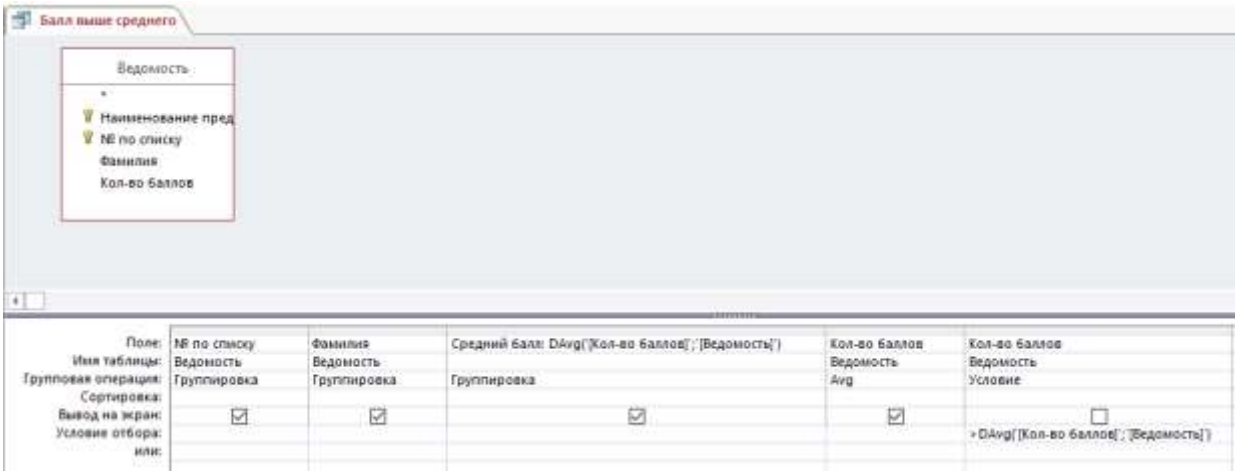
1. Визначимо середній бал кожного студента, використовуючи в групових операціях функцію Avg

2. Для поля «Кількість балів» використовуємо групову операцію Умови та Умови відбору будуть наступні:

>DAvg('[Кількість балів]';'[Відомість]')

3. Додамо поле, що додатково обчислюється, за визначенням середнього балу по групі:

Середній бал: **DAvg**('[Кількість балів]';'[Відомість]')



Результати виконання запиту:

Балл выше среднего

№ по списк	Фамилия	Средний балл по группе	Средний балл студента
1	Дяків	71,7692307692308	80,0
2	Жуковський	71,7692307692308	91,7
3	Ільясов	71,7692307692308	84,3
4	Кваша	71,7692307692308	90,0
5	Маловичко	71,7692307692308	85,0
6	Петлінський	71,7692307692308	100,0
7	Радалов	71,7692307692308	86,7
8	Серт	71,7692307692308	86,7
9	Тарасов	71,7692307692308	98,3
10	Халецька	71,7692307692308	86,7
11	Саввіч	71,7692307692308	100,0
12	Шувалов	71,7692307692308	93,3
13	Яворський	71,7692307692308	85,0

Приклад 3:

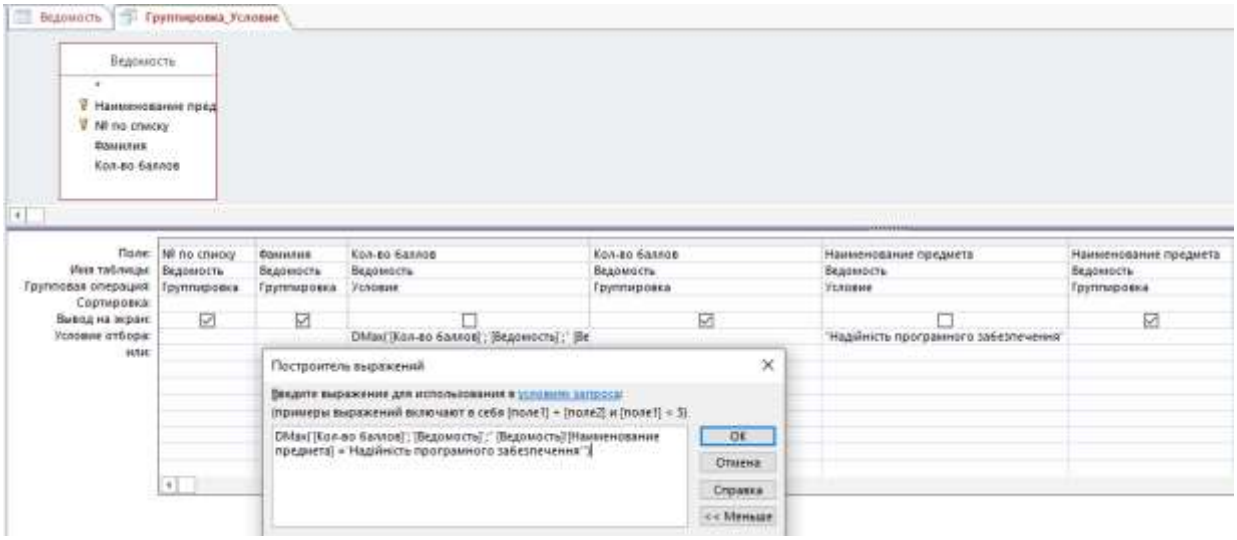
Потрібно вивести список студентів із максимальним балом на предмет 'Надійність програмного забезпечення'.

Для цього:

1. У групових операціяхдля поля «Найменування предмета» вибираємо Умову та умовою відбра ставимо предмет 'Надійність програмного забезпечення'.

2. В умовах відбору для поля«Кількість балів»додаємо вираз:

DMax('[Кількість балів]';'[Відомість]';" [Відомість]![Найменування предмета] ='Надійність програмного забезпечення'")



Результат виконання запиту:

Ведомость | Группировка_Условие

№ по списк	Фамилия	Кол-во баллов	Наименование предмета
2	Жуковський	90	Надійність програмного забезпечення
5	Маловичко	90	Надійність програмного забезпечення
9	Тарасов	90	Надійність програмного забезпечення
12	Шувалов	90	Надійність програмного забезпечення

3.9 ЗАПИТИ НА ЗМІНУ.

Досить часто при роботі з даними виникає потреба в їхній модифікації,

Модифікацію невеликого обсягу даних можна зробити, як зазначалося вище, вручну як Таблиці.

Модифікація великої кількості записів повинна проводитися тільки за допомогою спеціальних засобів.

У сучасних СУБД модифікацію великих обсягів даних можна зробити за допомогою запитів на зміну.

Запит на зміну - Це запит, який за одну операцію вносить зміни до декількох записів або створює в базі даних нову таблицю.

У QBE СУБД MS Access можна створити чотири типи запитів зміну: запит створення таблиці, запит оновлення записів, запит додавання записів і запит видалення записів (рис.3.1). У вікні бази даних кожен тип запиту ідентифікується власною піктограмою.

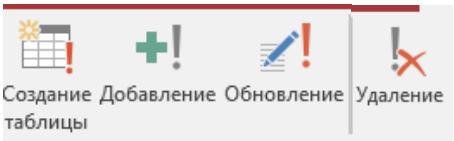


Рис.3.1. Типи запитів на зміну

3.9.1 ЗАПИТ СТВОРЕННЯ ТАБЛИЦІ

Запит створення таблиці дозволяє створити нову таблицю на основі всіх або частин даних з однієї або декількох таблиць.

Запит створення таблиці корисний до виконання таких действий:

- Створення таблиці для експорту в іншу БД;

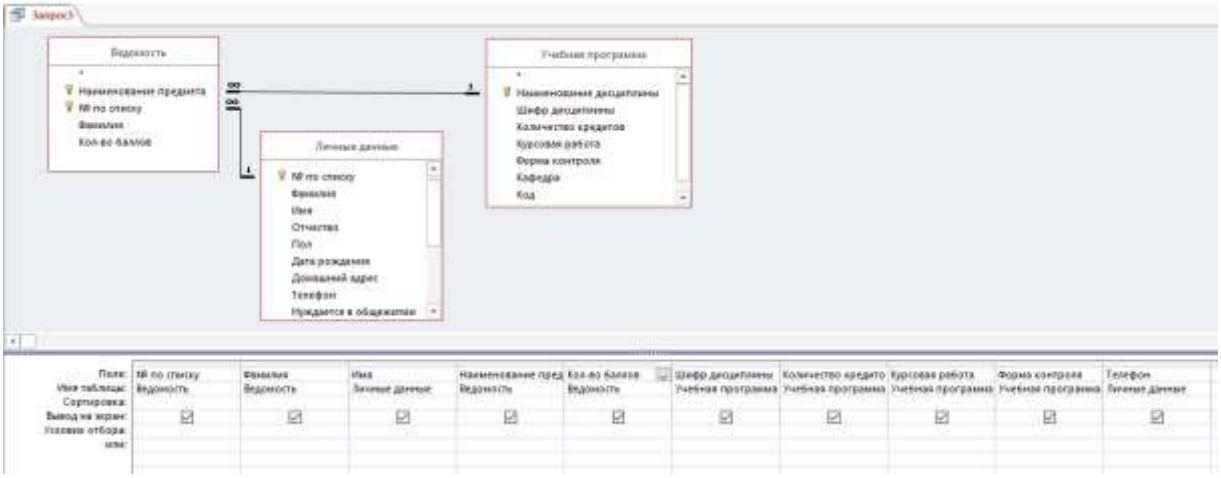
- Створення резервної копії таблиці;

- Створення архівної таблиці, що містить старі записи.

Послідовність створення запиту:

1. Створити запит на вибірку з умовою відбору, що дозволяє знайти всі записи, які будуть у новій таблиці.

2. Виконати запит на вибірку та перевірити правильність відбору записів (якщо записи не відповідають критеріям відбору, слід перейти в режим Конструктора та вказати коректні умови відбору).



3. Запит на вибірку перетворити на запит створення таблиці. Для цього в меню коструктора виберіть тип запиту Створення таблиці.

4. Для виконання запиту створення таблиці вибрати команду Запит | Запуск або клацнути на кнопку Запуск на панелі інструментів Конструктор запитів.

З'явиться діалогове вікно Створення таблиці (рис.3.2), в якому необхідно вказати ім'я таблиці, що створюється, і місце її розміщення.

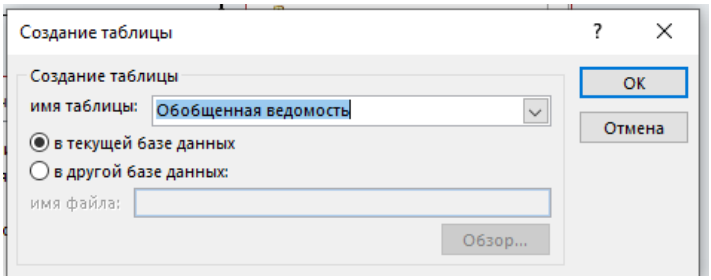


Рис.3.2. Діалогове вікно Створення таблиці

Після запуску виконання запиту створення таблиці на екрані з'являється діалогове вікно на підтвердження операції створення таблиці та додавання до неї записів (рис.3.3).

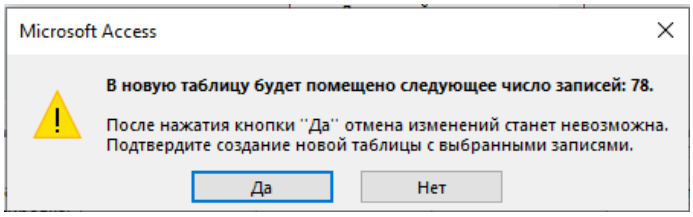


Рис.3.3. Вікно підтвердження операції створення таблиці

Після створення таблиці її можна відкрити в режимі Конструктор та змінити імена полів, типи даних, настроїти властивості полів, вказати ключові поля, створити індекси тощо.

3.9.2 ЗАПИТ НА ОНОВЛЕННЯ ГРУПИ ЗАПИСІВ

Запит на оновлення дозволяє внести зміни до групи записів існуючої таблиці.

За допомогою запиту на оновлення можна за один раз змінити значення кількох полів, увімкнувши їх у бланк запиту та визначивши вирази, які будуть використовуватись для оновлення цих полів.

Послідовність створення запиту на оновлення:

1. Створити запит на вибірку з умовою відбору, що дозволяє знайти всі записи, що підлягають оновленню (рис.3.4).

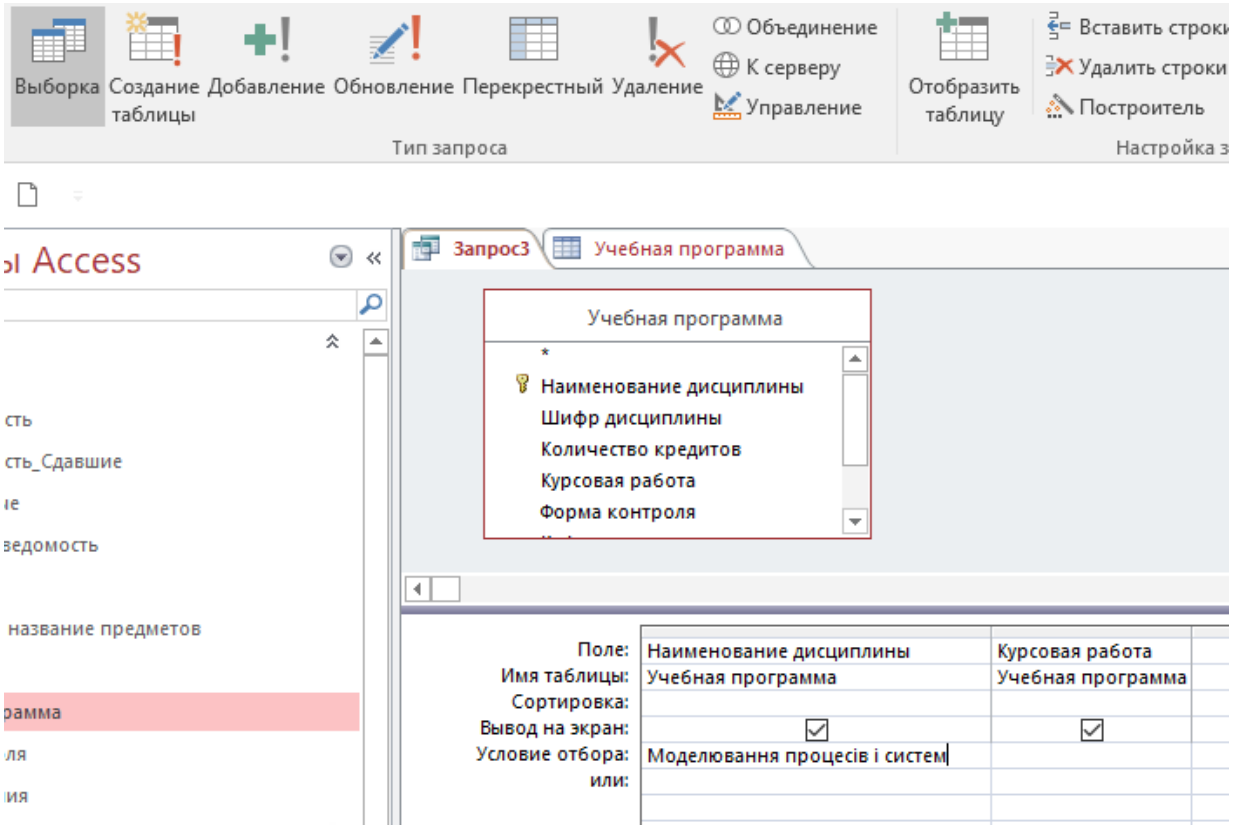


Рис.3.4. Запит на вибірку в режимі Конструктор

2. Виконати запит на вибірку та перевірити правильність відбору записів (якщо записи не відповідають критеріям відбору, слід перейти в режим Конструктора та вказати коректні умови відбору).

3. Запит на вибірку перетворити на запит на оновлення. Цю операцію можна виконати:

– використовуючи меню Запит | Оновлення;

– клацнувши на кнопку Тип запиту та зі списку, що розкривається, вибравши команду Оновлення.

При подібному перетворенні СУБД MS Access змінює заголовок вікна запиту та додає до бланку QBE рядок Оновлення.

4. У рядку Оновлення ввести нові значення (або вирази) для полів, дані яких повинні бути змінені в відібраних записах (Рис.3.5).

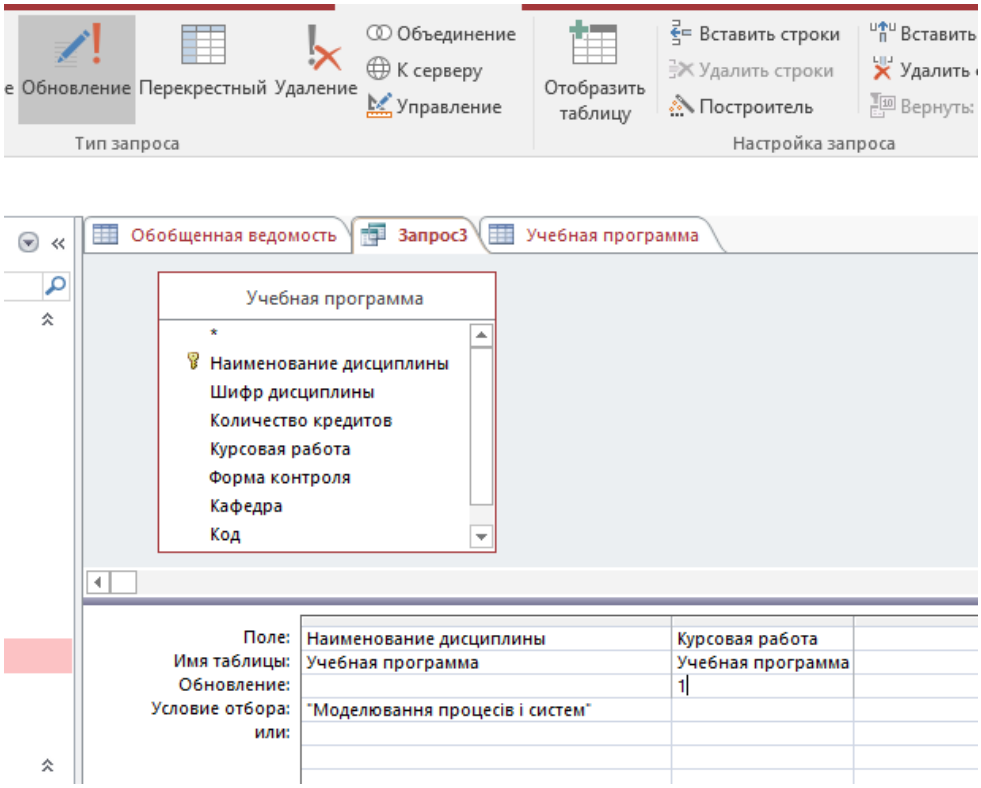


Рис.3.5. Запит на оновлення у режимі Конструктор

5. Для виконання запиту на оновлення виберіть команду Запит | Запуск або клацнути на кнопку Запуск на панелі інструментів Конструктор запитів. СУБД MS Access визначить кількість записів, що оновлюються, а потім виведе діалогове вікно для підтвердження операції оновлення (Рис.3.6).

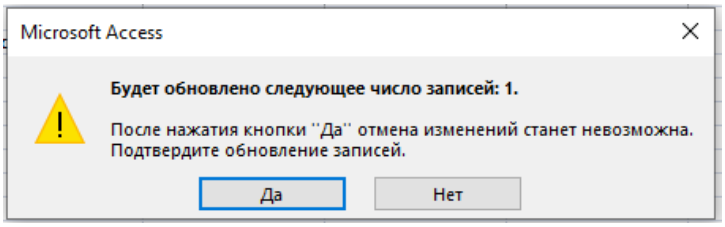


Рис.3.6. Вікно підтвердження операції поновлення

6. Після клацання по кнопці Так відбудеться оновлення вказаної кількості записів і скасування проведених змін стане неможливим. Тому при зміні даних за допомогою запиту на зміну необхідно дотримуватися правила "сім разів відміряй і один раз відріж" (Рис.3.7).

Наименование дисциплины	Шифр дисці	Количество кредитов	Курсовая ра	Форма конт	Кафедра
Автоматизація виробничих процесів	ПП Н.10	4,5	☐	0	УСБЖД
Безпека життєдіяльності та основи хорони прац	ПП Н.09	3,0	☐	0	КТА
Моделювання процесів і систем	ПП Н.13	4,0	☑	1	КТА
Надійність програмного забезпечення	ПП В.28	4,5	☑	1	КТА
Основи комп`ютерно інтегрованого управління	ПП Н.12	4,0	☐	0	КТА
Прикладна метематика	ПП В.29	4,5	☑	0	КТА
		0,0	☐		

Рис.3.7. Результат виконання запиту.

7. Для подальшого використання зберегти запит у базі даних, надавши йому ім'я (при необхідності).

3.9.3 ЗАПИТ НА ДОДАВАННЯ ЗАПИСІВ.

Запит на додавання дозволяє додати групу записів з однієї або кількох таблиць до кінця існуючої таблиці.

Наприклад, ми хочемо створити копію таблиці Відомість, куди поміщатимемо записи про студентів, які склали залік або іспит (тобто мають >=60 балів).

Створимо копію таблиці Відомість. У цьому збережемо лише її структуру(Рис.3.8.)

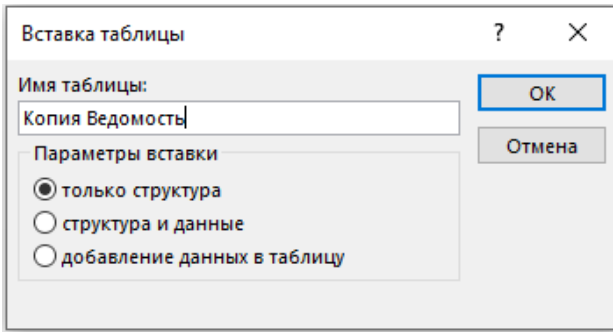


Рис3.8. Вставлення таблиці.

Створимо запит на вибірку на основі таблиці Відомість з необхідною умовою відбору.

Послідовність створення запиту на додавання практично така сама, як при створенні запиту на оновлення.

Різниця полягає тільки в тому, що при зміні типу запиту (з запиту на вибірку в запит на додавання записів) на екрані з'явиться діалогове вікно Додавання, в якому необхідно вказати ім'я таблиці для прийому записів, що додаються, і місце її розміщення, а у вікні Конструктора з'явиться рядок Додавання, в якій для полів вихідної таблиці необхідно встановити відповідні імена полів таблиці-приймача (рис.3.9)

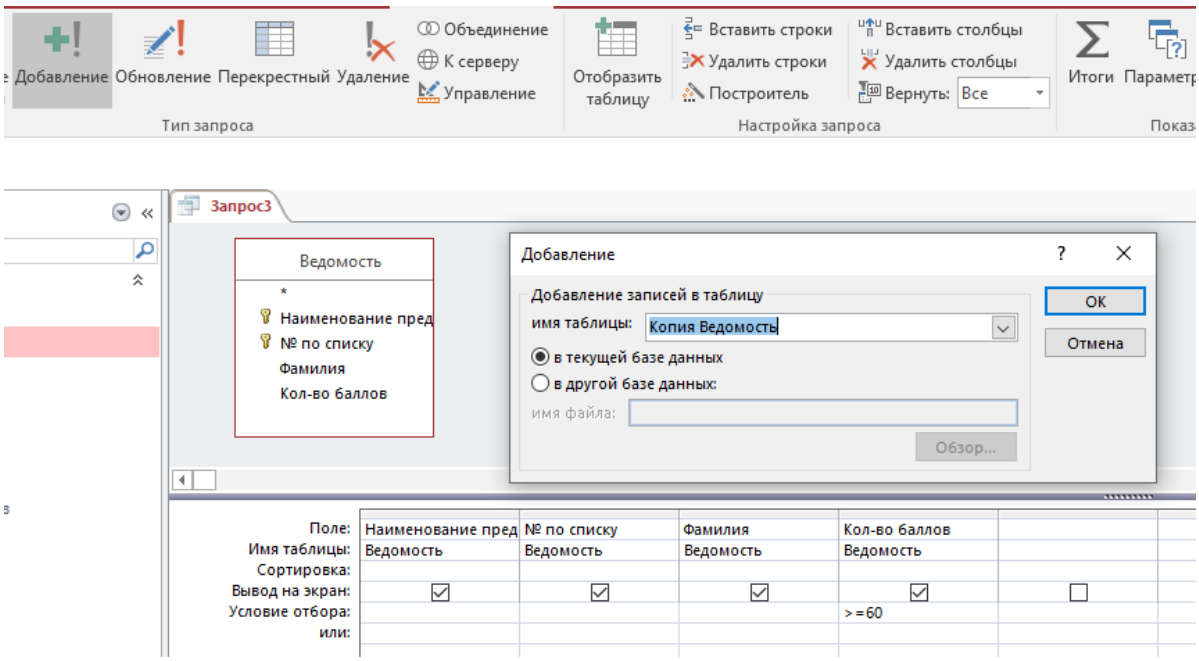


Рис.3.9. Запит на додавання в режимі Конструктор

Після запуску виконання запиту додавання записів на екрані з'являється діалогове вікно на підтвердження операції додавання (рис.3.10).

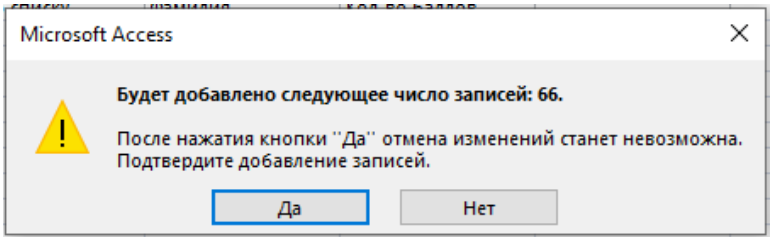


Рис.3.10. Вікно підтвердження операції видалення

3.9.4 ЗАПИТ НА ВИДАЛЕННЯ ЗАПИСІВ.

Запит на видалення записів дозволяє видалити групу записів із таблиці. За допомогою запиту на видалення можна видаляти лише весь запис, а не окремі поля.

Послідовність створення запиту на видалення записів така сама, як і під час створення запиту на оновлення.

Різниця полягає тільки в тому, що у вікні Конструктора з'явиться рядок Видалення (рис.3.11).

Наприклад, в Узагальненій таблиці ми хочемо залишити лише інформацію про іспити (тобто видалимо записи з формою контролю = 0).

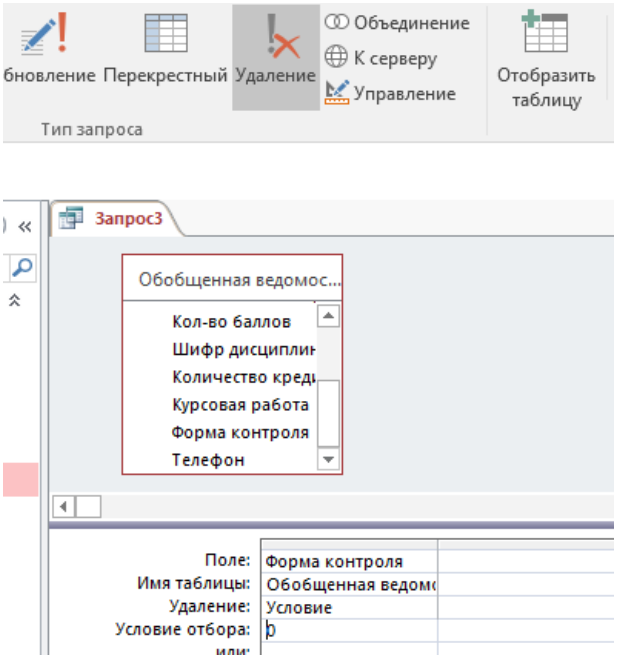


Рис.3.11. Запит на видалення в режимі Конструктор

Після запуску виконання запиту видалення записів на екрані з'являється діалогове вікно на підтвердження операції видалення (рис.3.12).

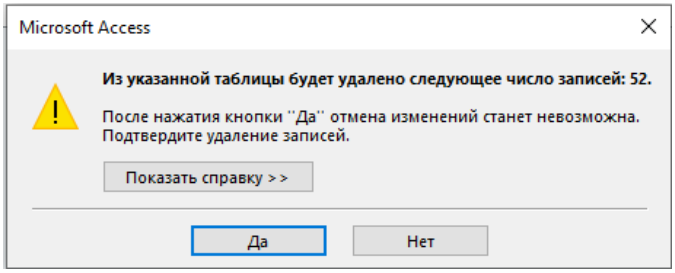


Рис.3.12. Вікно підтвердження операції видалення

4 ФОРМИ.

4.1 ЗАГАЛЬНІ ВІДОМОСТІ.

Форми призначені для перегляду, введення, редагування та управління даними. З використанням форми Access надає користувачеві значно більше можливостей до роботи з інформацією проти її стандартним поданням у режимі таблиці. Зручність застосування форм для роботи з даними полягає в наступному:

- Форми зазвичай дозволяють відобразити на екрані весь запис повністю, причому порядок прямування полів може бути змінений, а частина з них взагалі не включена у форму.
- У формах крім редагованих полів, що містять дані з таблиць БД, можна розміщувати і поля, що не редагуються (обчислюються).
- До форм можна додавати коментарі, малюнки, графіки, змінювати їх зовнішній вигляд, підбираючи відповідні шрифти, фон та стиль оформлення.
- Форми дозволяють спростити, а часто повністю автоматизувати введення нових даних.
- У формах можна розміщувати різні кнопки (кнопкові форми), натискання на які призводить до відкриття інших форм, виконання запитів, друку звітів і т.д.
- Форми можуть використовуватися як діалогові вікна і містити власне меню користувача.

Основним *джерелом даних* дляФорми є таблиці та запити. Інформація може бути включена у форму результаті імпорту різних об'єктів чи проведення обчислень. Якщо через форму здійснюється оновлення даних, вони оновлюються й у джерелі.

Побудоваформи – ітеративний процес. Після створення макета форми потрібно переглянути його, щоб переконатися у його придатності. Якщо потрібно змінити, можна повернутися до коригування макета.

Для створення макетаформи потрібно у вікні БД перейти на вкладку **Форми** та натиснути клавішу Створити. Після вибору джерела даних для форми, а також способу її створення при переході до наступного етапу автоматично оновлюються панелі інструментів. З'являється панель інструментів. Конструктор форм.

Створення, коригування та перегляд форми здійснюється в різних режимах:

- у режимі конструктора форма створюється та коригується;
- у режимі форми або режим таблиці форма використовується для роботи з даними;
- у режимі попереднього перегляду (режим макету) форма переглядається перед друком.

4.2 СТРУКТУРА ФОРМИ.

Форма складається з кількох розділів (рис. 4.1), причому обов'язковим є лише один із них — область даних. Крім цього розділу у формі можуть бути такі розділи: заголовок та примітка форми, а також верхній та нижній колонтитули. Інформація в них запроваджується розробником форми. Ці розділи мають такі призначення:

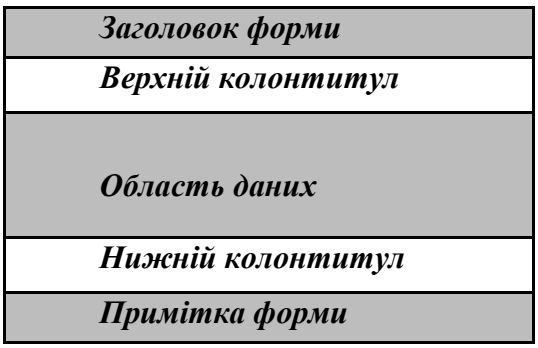


Рис. 4.1. Структура форми

Заголовок форми може містити назву форми, інструкції роботи з нею, а також іншу інформацію (поточну дату, час і т.д.). У режимі форми він знаходиться у верхній частині екрана, а під час друку – у верхній частині першої сторінки. У режимі таблиці цей розділ не відображається.

Верхній колонтитул може містити заголовки та будь-які інші відомості, що містяться у верхній частині кожної сторінки форми під час друку.

Нижній колонтитул також використовується під час друкута може містити номер сторінки, дату та іншу інформацію.

Примітка форми може містити інструкції щодо використання форми, кнопки та вільні елементи керування для введення або виведення даних. У режимі форми область приміток знаходиться в нижній частині екрана, а друкується на останній сторінці.

При створенні форми за допомогою конструктора вона містить лише один розділ - область даних. Інші розділи можна додати, використовуючи відповідні пункти контекстного меню форми або меню Вид.

Форма може включати підлеглі форми, всередині яких можуть відображатися три розділи: заголовок, область даних та примітка.

Які всі об'єкти БД, форма та її розділи мають властивості. У властивості задаються режими роботи з формою, вид форми, джерело даних, використання фільтра. Властивості розділів використовуються при виведенні на екран або друк.

4.3 ВИДИ ФОРМ.

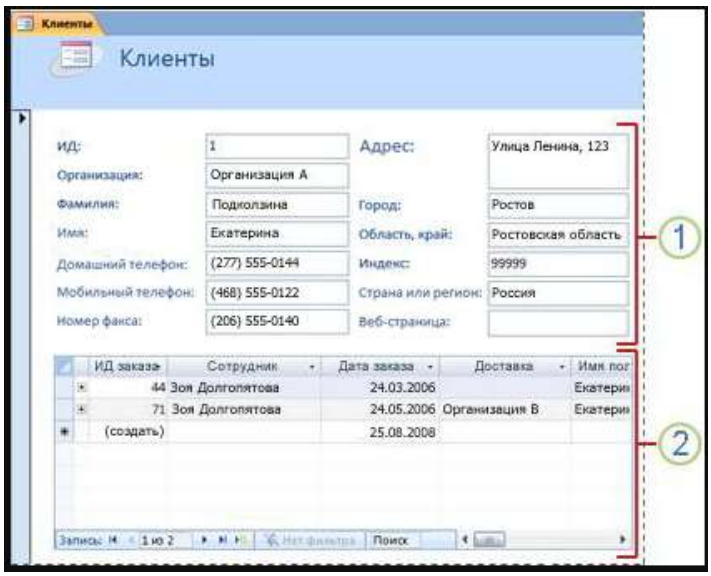
4.3.1 ФОРМА ДЛЯ ВВЕДЕННЯ ТА МОДИФІКАЦІЇ ДАНИХ.

4.3.1.1. ФОРМА З ОДНИМ ЕЛЕМЕНТОМ.

Засіб «Форма» можна використовувати для створення форми з одним елементом. Форма такого типу одночасно відображає відомості лише про один запис, як показано на малюнку нижче.

1. Ця форма відображає інформацію лише для одного запису.

2. У деяких випадках Access додає підтаблицю, яка містить пов'язані із записом відомості.



Створення форми з одним елементом:

1. У області навігації виберіть таблицю або запит із даними, які потрібно додати до форми.

2. На вкладці Створення у групі Форми виберіть команду Форма.

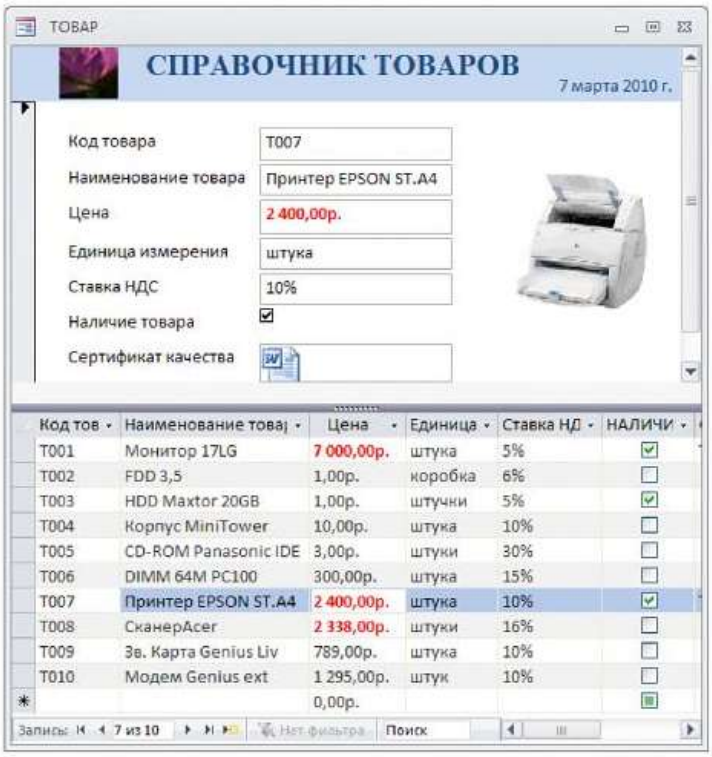
Access створить форму та відобразить її в режимі макету. У цьому режимі у форму можна вносити зміни, але вона продовжує відображати дані. Наприклад, можна змінити розмір текстових полів, щоб помістити всі дані.

3. Щоб розпочати роботу з формою, перейдіть до режиму форми.

- На вкладці Головна виберіть пункт Вигляд, а потім — Режим форми.

4.3.1.2. РОЗДІЛЕНА ФОРМА.

Даний вид форми дозволяє одночасно відображати дані у двох уявленнях - в режимі форми та в режимі таблиці.



Створення нової розділеної форми за допомогою інструмента «Розділена форма»:

Ця процедура створює нову розділену форму з нуля. Форма створюється на основі таблиці або запиту, які вибираються в області переходів або відкриті у режимі таблиці.

1. В області переходів клацніть таблицю або запит із даними, які мають відображатися у формі, або відкрийте таблицю або запит у режимі таблиці.

2. На вкладці Створення в групі Інші форми клацніть Розділена форма.

Буде створено нову форму та відображено в режимі макету. У режимі макету можна внести зміни до структури форми при одночасному відображенні даних. Наприклад, за потреби можна налаштувати розмір полів відповідно до даних.

Перетворення наявної форми на розділену форму:

Наявну форму можна перетворити на розділену форму, задавши кілька властивостей форми:

1. Відкрийте форму в режимі конструктора, клацнувши її правою кнопкою миші в області переходів та вибравши команду Конструктор.

2. Щоб відкрити вікно властивостей, якщо воно не відкрите, натисніть клавішу F4.

3. У списку у верхній частині вікна властивостей виберіть пункт Форма.

4. У вікні властивостей на вкладці Формат у списку Режим за замовчуванням виберіть пункт Поділ форми.

5. Перевірте форму у режимі форми. Щоб перейти до режиму форми, двічі клацніть ім'я форми в області переходів.

4.3.1.3. ФОРМА ДЛЯ КІЛЬКОХ ЕЛЕМЕНТІВ (СТРІЧКОВА ФОРМА).

Створення форми за допомогою засобу "Кілька елементів"

Клиенты

Клиенты

ИД	Организация	Фамилия	Имя	Должность
1	Организация А	Подколзина	Екатерина	Владелец
2	Организация Б	Кирилов	Антон	Владелец
3	Организация В	Орехов	Алексей	Уполномоченн
4	Организация Г	Маринова	Надежда	Руководитель
5	Организация Д	Грачев	Николай	Владелец
6	Организация Е	Куликов	Евгений	Руководитель
7	Организация Ж	Нагайчук	Кирилл	Владелец
8	Организация З	Ожогина	Инна	Уполномоченн
9	Организация И	Климов	Сергей	Руководитель
10	Организация К	Димитров	Борислав	Руководитель

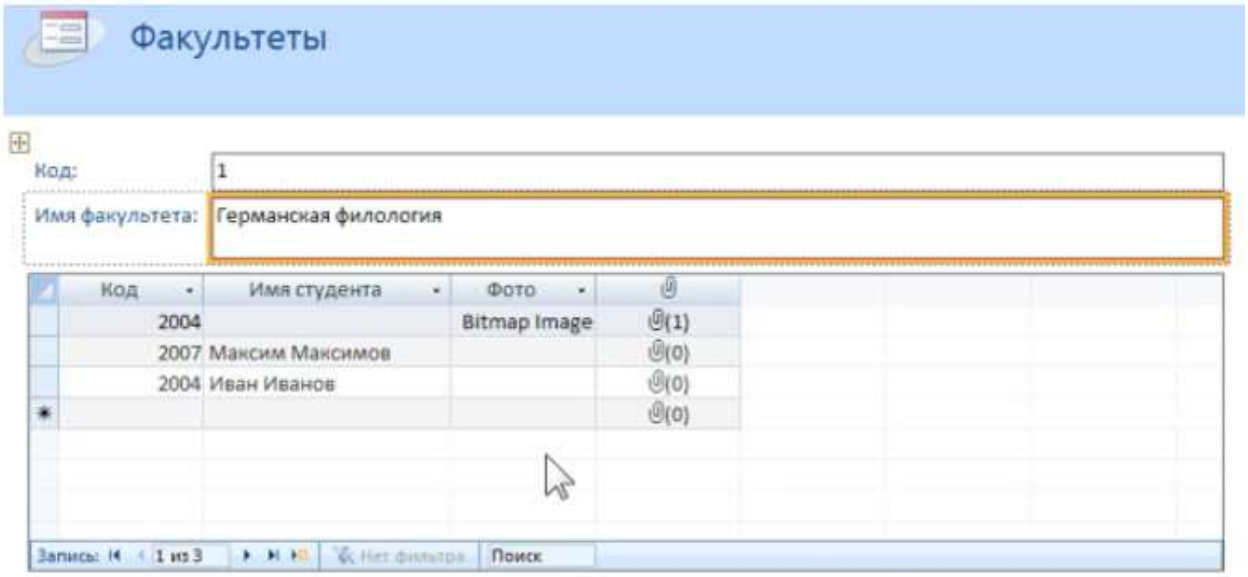
Дані відображаються у вигляді рядків та стовпців, причому одночасно відображаються кілька записів. Однак це форма, і тому її можна більш тонко налаштувати, ніж таблицю. До неї можна додавати графічні елементи, кнопки та інші елементи керування.

Створення форми для кількох елементів:

1. На панелі навігації виберіть таблицю або запит із даними, які потрібно розмістити у формі.
2. На вкладці Створення в групі Інші форми клацніть Кілька елементів.
3. Щоб розпочати роботу з формою, перейдіть до Режиму форми.
4. На вкладці Головна виберіть пункт Вигляд, а потім — Режим форми.

4.3.1.4. СКЛАДОВА ФОРМА (ГОЛОВНА І ПІДЛЕГЛА, З СТАВЛЕННЯМ «ОДИН-БАГАТЬОМ»).

Існує спосіб, який прискорює створення складової форми. Спочатку виберіть на створеній порожній новій формі стрічку Створення > Форми > Конструктор.



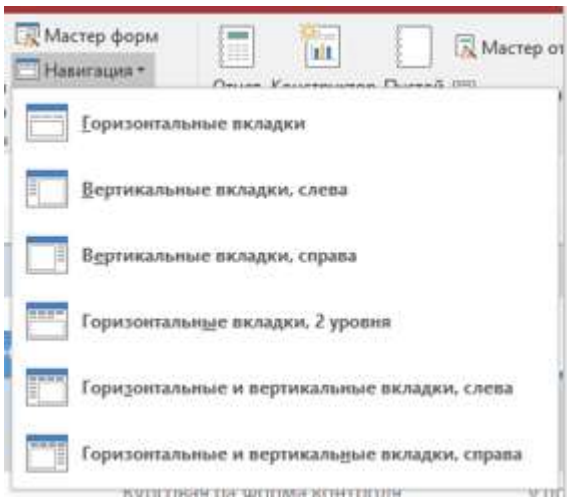
Знайдіть форму, яку хочете використовувати у підлеглій формі, та перетягніть її мишею з області переходів на робочу поверхню вашої нової форми. Програма Access створить елемент керування Підпорядкована форма, де відображається ця форма. Можна також перетягнути таблицю на вашу форму, у разі Access створить підлеглу форму для таблиці (і попросить вибрати неї ім'я).

Ця складова форма- пульт управління «все в одному» для додавання та перегляду товарів та перегляду списку клієнтів. Підготовлені та включені до складу програми Access шаблони часто використовують складові форми для розміщення кількох пов'язаних завдань редагування в одному місці

4.3.2 ЗВЕДЕНА ТАБЛИЦЯ (ЗВЕДЕНА ДІАГРАМА). (В Access 2013 цей режим вже не підтримується. Це завдання можна вирішити в Excel, імпортуючи дані з Access).

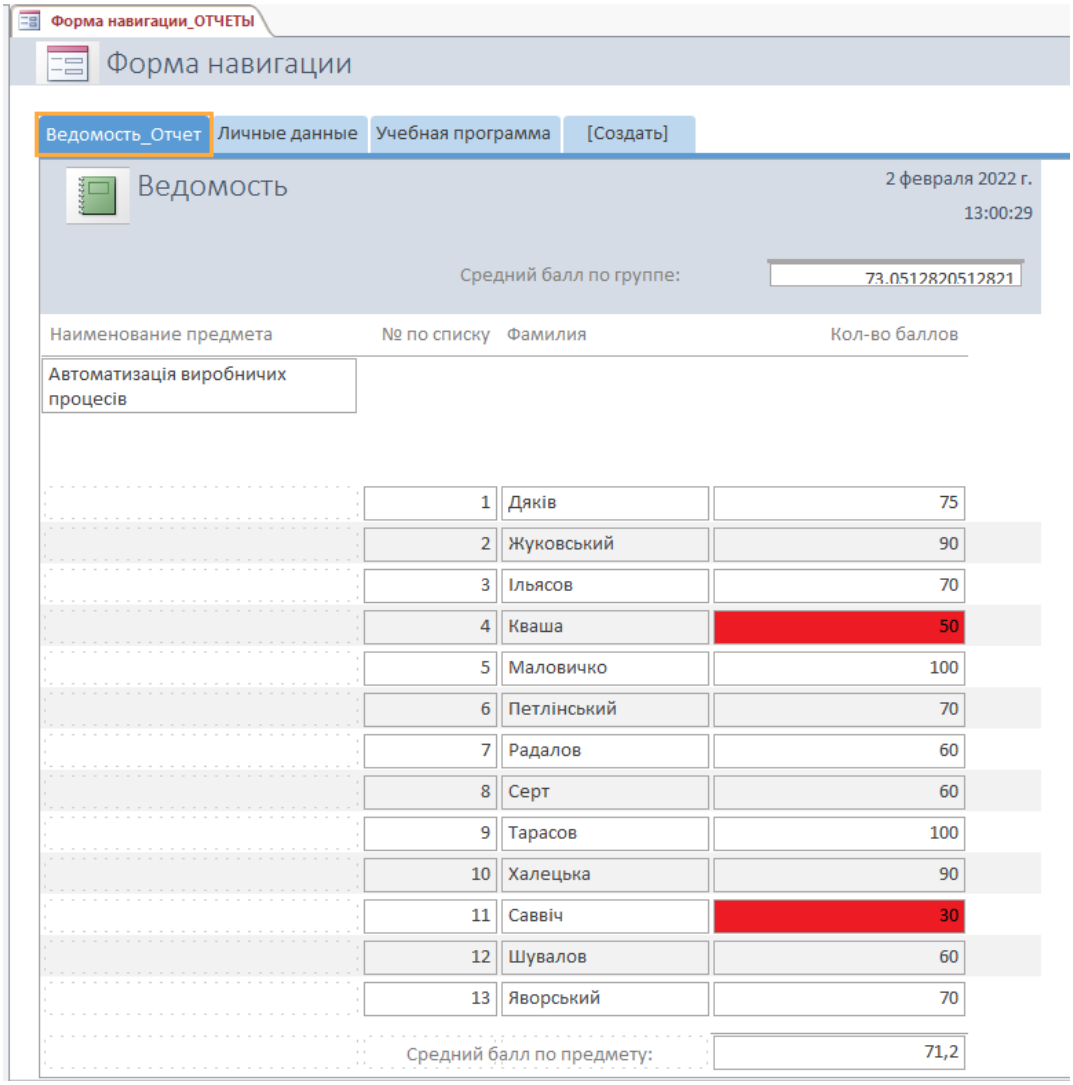
4.3.3. ФОРМА НАВІГАЦІЇ.

На вкладці Створення у групі Форми виберіть Навігація, а потім виберіть потрібний стиль форми навігації.



Access створить форму та відобразить її в режимі макету.

Для відображення об'єкта достатньо, не виходячи з Макету, перетягнути його на вкладку Форми навігації.



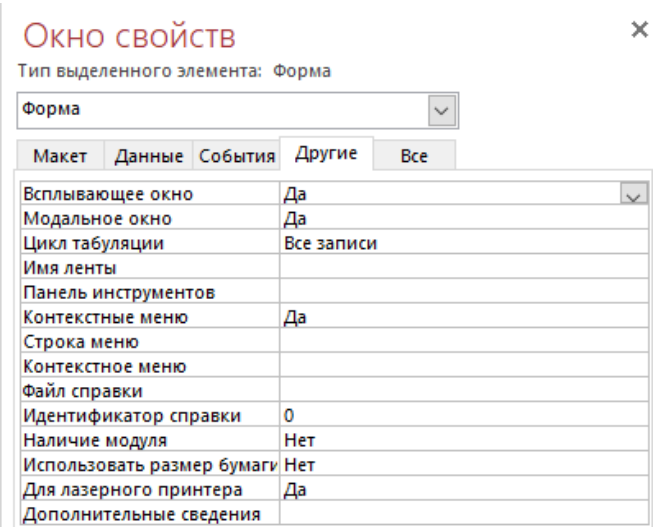
4.3.4 ВИПЛИВАЮЧІ ФОРМИ ТА ДІАЛОГОВІ ВІКНА.

У Access існують такі вікна, які розміщуються на екрані поверх решти всіх вікон, навіть якщо в даний момент активним є інше вікно. Приклад такого вікна є вікно

Властивості в Конструкторі форм. Ви також можете у своїй програмі створювати форми такого типу. Вони називаються формами, що спливають.

Наприклад, якщо головна кнопкова форма, з якої можна виконати основні дії в програмі, невелика і не закриває інші форми, її можна зробити спливаючою, щоб вона була доступна будь-коли.

Щоб форма мала таку властивість, необхідно присвоїти значення та властивості *спливаюче вікно* . Ця властивість знаходиться на вкладці Інші вікна властивостей форми.



Звичайну форму також можна перетворити на *модальне діалогове вікно*, якщо властивості Модальне вікно цієї форми задати значення Так. Модальна форма відрізняється тим, що поки вона відкрита, ви не можете перемістити фокус на інший об'єкт — форму, меню, кнопку панелі інструментів тощо, тобто інші об'єкти стають недоступними, поки не буде закінчено роботу з цією формою і вона не буде закрито.

Модальні форми зазвичай використовуються для створення спеціальних діалогових вікон, які запитують користувача певну інформацію. Щоб виконувати інші завдання, користувач повинен ввести цю інформацію та закрити форму. Зазвичай у такому вікні встановлюють властивості Кнопка віконного меню, Кнопки розмірів вікна та Кнопка закриття) таким чином, що ці кнопки не відображалися у формі. Зате у самій формі створюють дві кнопки ОК та Скасувати. Кнопка ОК дозволяє виконати подальші дії та закриває форму. Кнопка Скасування закриває форму і виконує всі дії з переривання операції, що виконується. Для цього створюють програми VBA або макроси, які пов'язують з подією натискання кнопки кожної з цих кнопок.

4.4 СТВОРЕННЯ ФОРМ.

Форму можна створювати за допомогою безлічі засобів, що знаходяться на вкладці Створення у групі Форми (рис.4.2).

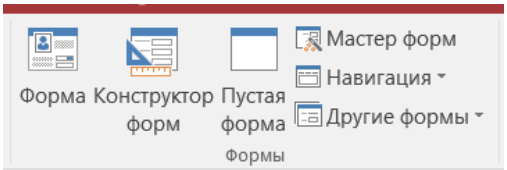


Рис. 4.2. Група Форми

Різні видиформ створюються зазвичай майстром форм чи користувачем з допомогою конструктора.

4.4.1 РОЗРОБКА АВТОФОРМ.

Можливе створення видів автоформ: в один стовпець, табличної стрічкової та вирівняної форми. Під час створення автоформи всі поля таблиці або запиту виводяться у форму автоматично.

*Форма в один стовпець*відображає поля, розташованів один стовпець. На екрані з'являється один запис.

*Таблична форма*відображає дані у вигляді кількох рядків та стовпців. Одночасно з'являється кілька записів. На вигляд і способів переміщення по записах вона нічим не відрізняється від звичайної таблиці Access.

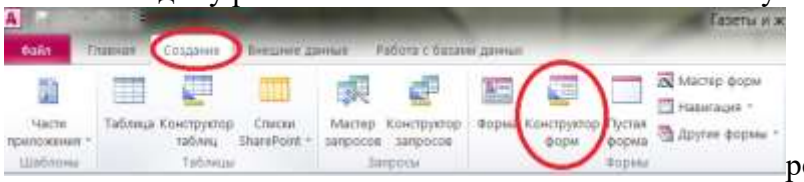
*Стрічковаформа*схожа на табличнуформу та відрізняється лише зовнішнім оформленням.

На відміну від автоформ при використанні майстра форм поля у форму можна вибирати. Майстер може створити крім перерахованих вище форм складову форму. Вона складається з головної форми та підлеглої. У цих формах відображаються дані з різних таблиць, причому ці таблиці найчастіше пов'язані ставленням «один-багатьом». Дані головної форми відображаються в один стовпець, а підлегла форма має табличний формат.

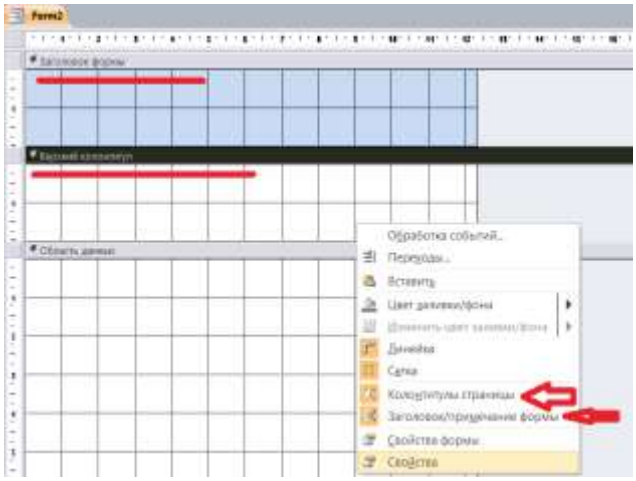
4.4.2 ВИКОРИСТАННЯ КОНСТРУКТОРА ФОРМ.

Щоб створити форму за допомогою інструмента Конструктор форм, виконайте такі дії:

1. На вкладці Створення натисніть кнопку Конструктор форм у групі Форми. В результаті виконаних дій у робочій області вікна з'явиться бланкпуста, яка не пов'язана з жодним дже

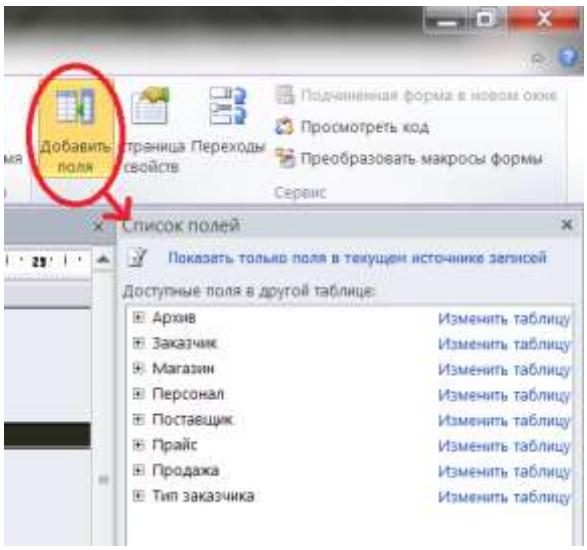


релом даних.



2. Зв'яжіть форму з джерелом даних. Для цього на панелі інструментів натисніть кнопку Властивості та виберіть джерело даних для форми.

3. Перетягніть поля для форми.



Створіть необхідний інтерфейс за допомогою представлених «кнопок»: 1- вибрати, 2- поле, 3- напис, 4 – кнопка, 5 – вкладка, 6 – гіперпосилання, .. 10 – розрив сторінки, 11 – список, 12 – діаграма, 19 - Вкладення, 20 - перемикач (інші).

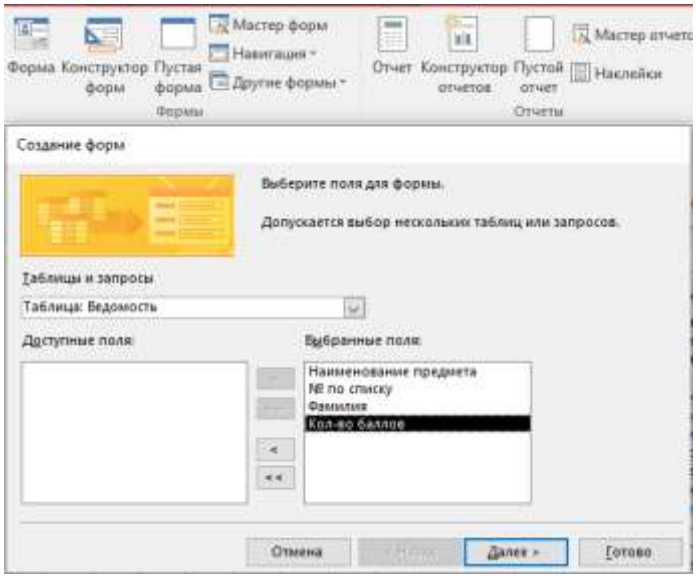
4.4.3 СТВОРЕННЯ ФОРМИ ЗА ДОПОМОГОЮ "МАЙСТРА ФОРМ".

Для отримання більшої свободи вибору полів, побудови форми, що відповідає структурним вимогам користувача, використовується режим Майстер форм, що відображаються на формі, замість згаданих вище інструментів можна скористатися майстром форм. Крім того, можна вказати спосіб угруповання та сортування даних, а також включити у форму поля з кількох таблиць або запитів за умови, що задані відносини між цими таблицями та запитами.

Для створення форми в режимі Майстра форм виконайте такі дії:

1. На вкладці Створення групи Форми натисніть кнопку Майстер форм.

2. Виберіть поля форми.



3. Виберіть зовнішній вигляд форми (в один стовпець, стрічковий, табличний, вирівняний).

4. Введіть назву форми.

5. Натисніть кнопку Готово (Далі).

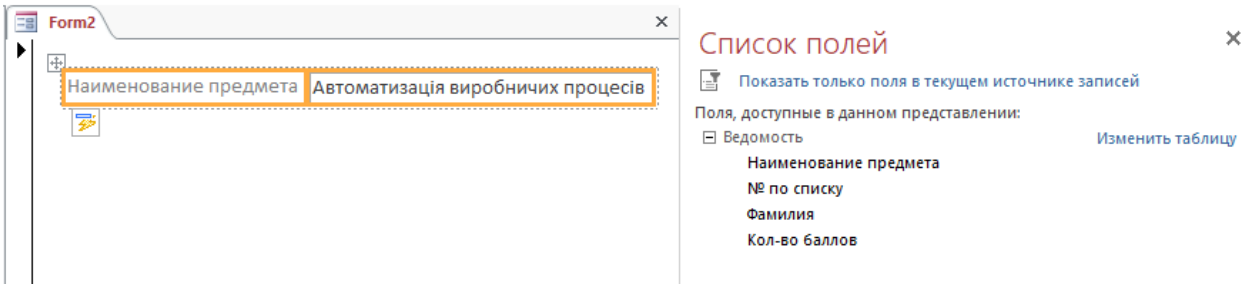
Отриману форму можна доопрацювати, використовуючи Конструктор форм.

4.4.4 СТВОРЕННЯ ФОРМИ ЗА ДОПОМОГОЮ ІНСТРУМЕНТУ ПУСТА ФОРМА.

Access 2010 дозволяє створювати форми у режимі порожньої форми. Цей режим дуже зручний оскільки дозволяє користувачеві створювати форми за необхідними параметрами, які можуть не підтримуватися в режимах Форма і Майстер форм.

Етапи створення форми в режимі Порожня форма:

1. На вкладці Створити у групі Форми натисніть кнопку Пуста форма. Access відкриє порожню форму в режимі макету та відобразить область Список полів.
2. В області Список полів клацніть знак "плюс" (+) поруч із таблицею або таблицями, що містять поля, які потрібно включити у форму.
3. Перетягніть поля, які необхідні для створення форми.



4. Відформатуйте, якщо потрібно, вид форми.

4.5 РОБОТА З ПАНЕЛЬЮ ЕЛЕМЕНТІВ.

Для додавання у форму елементів керування використовується панель елементів.

Елементи керування для введення, відображення і добору даних:

1. Написи - використовуються для відображення описового тексту.
2. Поля - використовуються для відображення даних із джерела записів.
3. Групи параметрів - використовуються для відображення обмеженого набору альтернатив.
4. Вимикачі, перемикачі, прапорці - використовуються для включення (вимикання) дії визначеної опції.
5. Списки - використовуються для відображення значень, одне з яких можна вибрати.
6. Поля зі списком - комбінація двох елементів: поля і списку, що розкривається.
7. Кнопки - використовуються для виконання дії або набору дій.
8. Підлеглі форми / звіти.

Елементи керування для виводу графіки й об'єктів:

1. Малюнки – вільний малюнок.
2. Вільні рамки об'єкта - використовуються для відображення об'єкта OLE, *не збереженого* в БД.
3. Зв'язані рамки об'єкта - використовуються для відображення об'єкта OLE *збереженого* в БД.

Елементи керування для організації даних:

1. Розриви сторінок
2. Вкладки
3. Лінії
4. Прямокутники

4.5.1 СТВОРЕННЯ ОСНОВНИХ ЕЛЕМЕНТІВ КЕРУВАННЯ.

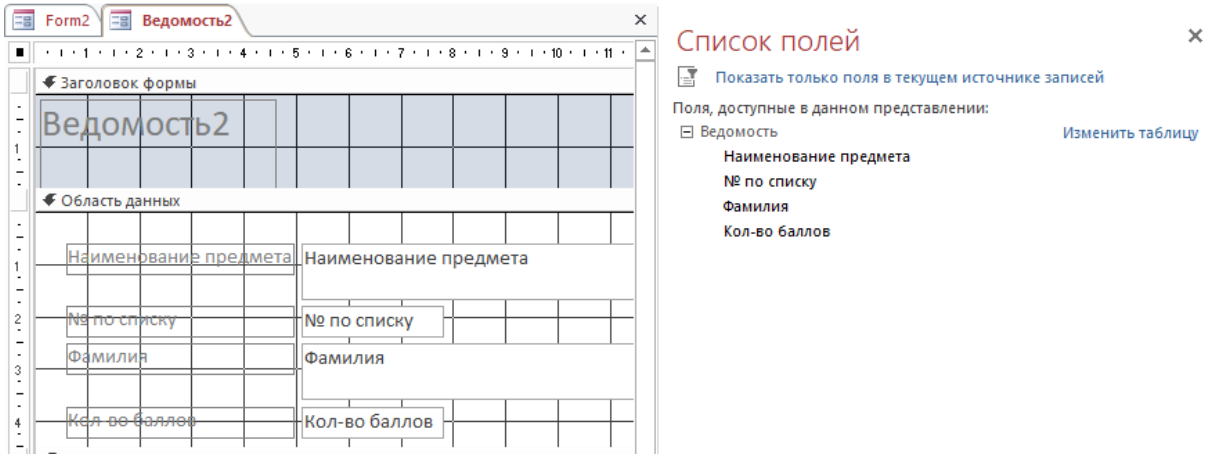
Елементи керування можуть бути *приєднаними*, *вільними* або *обчислюваними.* Кожному типу відповідає свій спосіб створення.

Для створення приєднаного елемента керування найпростіше використовувати список полів джерела даних, який зазвичай автоматично з'являється під час створення форми. Якщо на екрані його немає, його можна відкрити кнопкою Список полів або командою Список полів меню Вид.

Щоб створити приєднаний елемент керування потрібно вибрати пов'язане з ним поле в списку полів і, тримаючи натиснутою ліву кнопку миші, перетягнути його

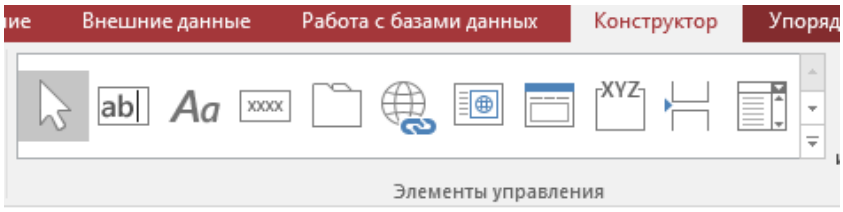
у потрібне місце форми. Access створить елемент керування, відповідний обраному полю, і задасть значення його властивостей, що відповідають типу даних та властивостям поля.

Можна перенести, попередньо виділивши, відразу кілька полів зі списку. І тут у формі з'явиться група елементів керування, приєднаних до цих полів. За бажання тип створеного елемента керування можна змінити.



Вільні елементи керування створюються за допомогою панелі елементів. Для

створення елемента потрібно натиснути відповідну кнопку на панелі, а потім клацнути мишею там області даних, де повинен знаходитися лівий верхній кут створюваного елемента керування стандартного розміру.

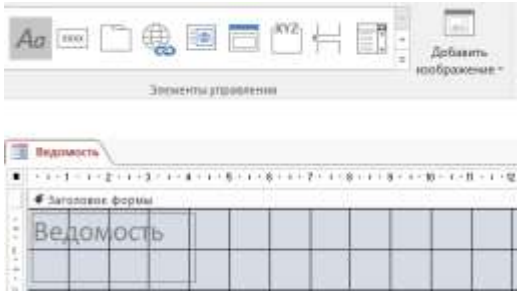


Зазвичай обчислюваним елементом керування є поле . 

4.5.1.1 СТВОРЕННЯ НАПИСУ.

Для створення напису, не пов'язаногоз яким елементом керування, потрібно натиснути кнопку Напис  вказати місце розташування напису, а потім ввести текст. Щоб перейти до напису на новий рядок, треба натиснути клавіші Ctrl+Enter. Для форматування вмісту напису можна використовувати кнопки формату тексту.

При зміні формату тексту напис може перестати розміщуватись у відведеній рамці. У цьому випадку потрібно привести у відповідність напис та розмір рамки. Для цього достатньо виділити напис та виконати команду За розміром даних у пункті Розмір меню Формат.



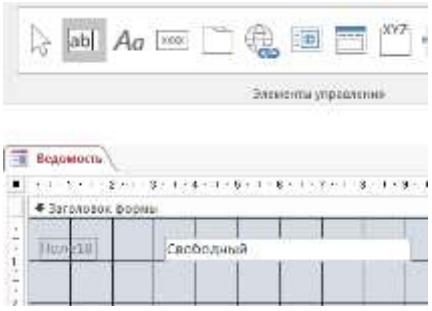
4.5.1.2 СТВОРЕННЯ ПОЛЯ.

Для створенняу формі приєднаного поля (тобто отримує інформацію з поля таблиці/запиту) найпростіше вибрати це поле у списку полів і перетягнути його в потрібне місце форми. У формі з'являться поле та його підпис. Підпис містить ім'я (або підпис, якщо він існує) вибраного поля таблиці/запиту. Саме поле у формі успадковує ім'я та властивості пов'язаного з ним поля. Це ім'я можна використовувати для посилання на значення поля у виразі.

Полета його підпис пов'язані між собою. Якщо спробувати виділити (клацнути мишкою) одне із них, виділяються обидва, причому маркери розміру і кнопка переміщення перебувають у тому елементі, у якому клацнули мишею. При переході в режим форми підпис залишається без змін, а у вікні поля з'являється інформація з пов'язаного поля джерела даних.

Можна перенестиу форму одночасно групу полів, попередньо виділивши їх, або всі поля (виділяються подвійним клацанням миші на заголовку Списку полів).

Для зміни властивостей поля можна викликати команду Властивості з контекстного меню або меню Вид. Властивості такожвикликаються кнопкою або подвійним клацанням миші.



4.5.1.3 СТВОРЕННЯ ПОЛЯ, ЩО ОБЧИСЛЮЄТЬСЯ.

Для створення елемента керування *Обчислюване поле* потрібно натиснути кнопку Поле панелі елементівта вказати місце розташування нового елемента. З'явиться Поле та пов'язаний з ним підпис. Потім потрібно створити вираз, що є джерелом даних цього поля.

Якщо вираз досить простий, тойого можна ввести у полі. Для цього потрібно клацнути мишкою всередині поля і потім ввести формулу обчислення значення поля, що містить знак рівності (=) і вираз, що обчислюється. У обчислюваного поля слід змінити напис. Можна також ввести формулу обчислення значення у комірку властивості Дані, відкривши вікно властивостей поля.

Якщо вираз має складну структуру, краще використовувати *Побудовник виразів*. Для цього потрібно натиснутимиші виділити поле, для якого створюється вираз, а потім викликати вікно його властивостей. Після натискання на клітинку властивості Дані з'явиться кнопка, клацання по якій викликає побудову виразів. Створений за його допомогою вираз з'явиться у вікні властивостей поля як значення властивості Дані. Потім можна настроїти формат виведення значення поля на екран за допомогою властивості Формат поля та, якщо потрібно, властивості Число десяткових знаків.

Приклад: На основі таблиці Навчальна програма створює форму. Додати обчислюване поле для підрахунку кількості годин (=[Кількість кредитів] *30), які відводяться на кожну дисципліну.

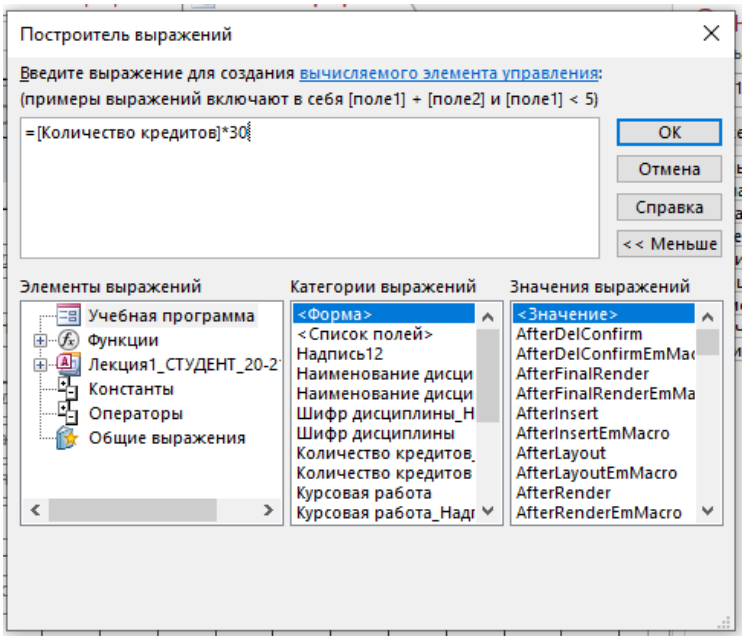


Рис. 4.1. Створення поля, що обчислюється, за допомогою Побудовника виразів.

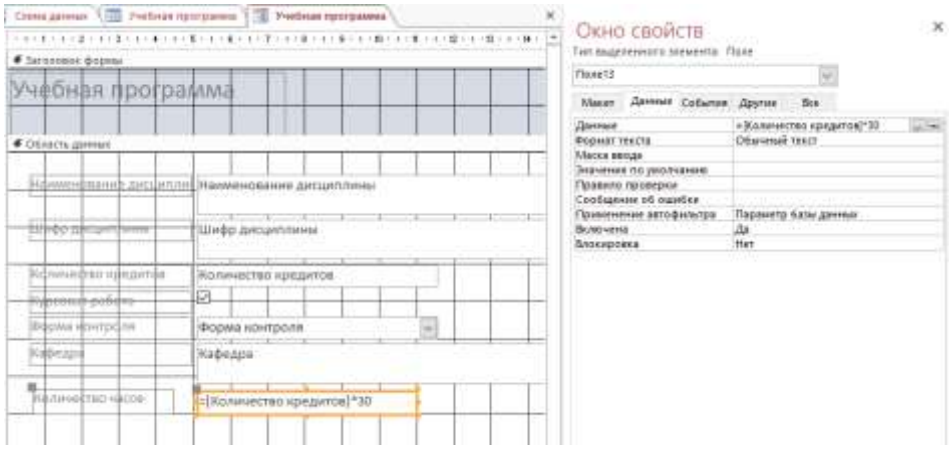


Рис. 4.2. Форма Навчальна програма у режимі конструктора.

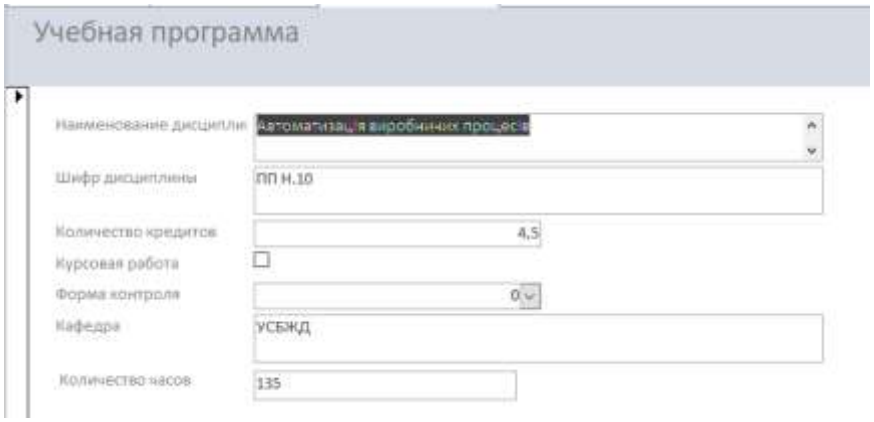


Рис. 4.3. Форма Навчальна програма у режимі форми

4.5.1.4 СТВОРЕННЯ СПИСКУ ЧИ ПОЛЯ ЗІ СПИСКОМ.

При введенні значення в поле таблиці або запиту через форму швидше та простіше вибрати потрібне значення зі списку, ніж вводити його з клавіатури. У цьому виключаються помилки введення. Існують два елемента керування, за допомогою яких можна організувати список значень, що прокручується, — поле зі списком і список.

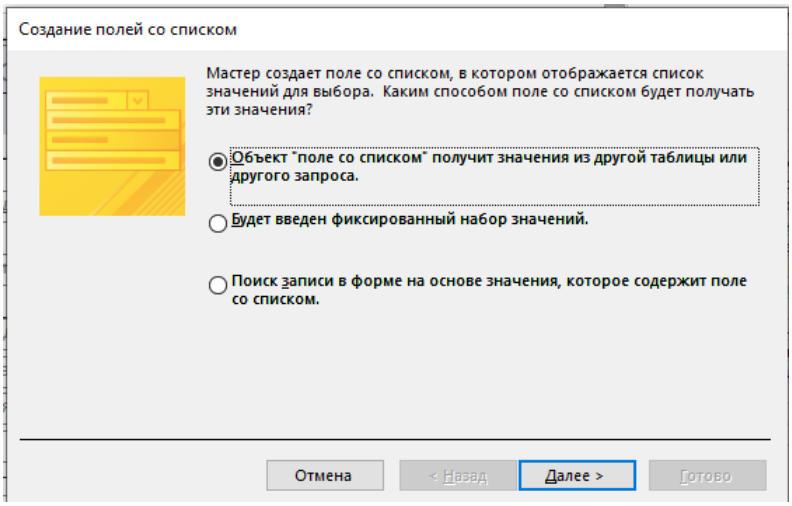
Список зображується у формі постійно, як значення, що вводиться може бути вибрано тільки значення зі списку.

У полі зі списком зображення списку з'являється лише при натисканні на кнопці, розміщеній у правому кінці поля. Крім того, користувач може дозволити введення нових значень, які не містяться у полі зі списком.

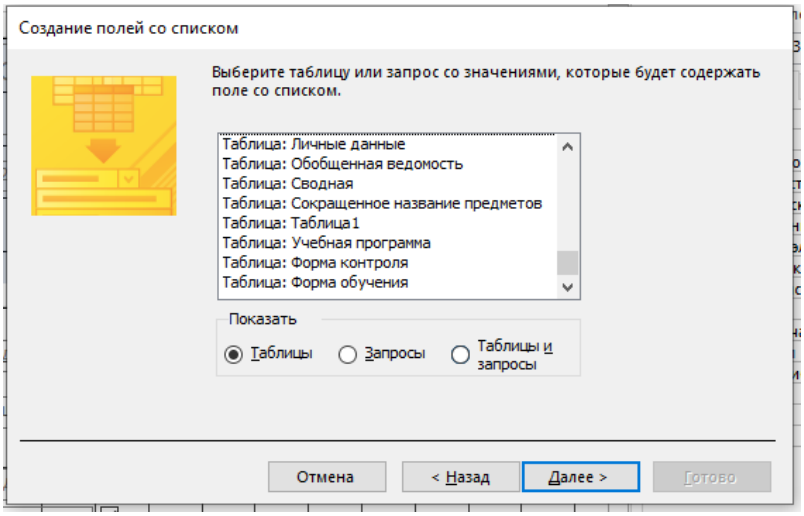
Для створення цих елементів керування потрібно натиснути кнопку **Список** або кнопку **Поле зі списком** на панелі елементів. Щоб викликати майстра, натисніть кнопку Використання майстра.

Робота майстра відбувається у кілька етапів.

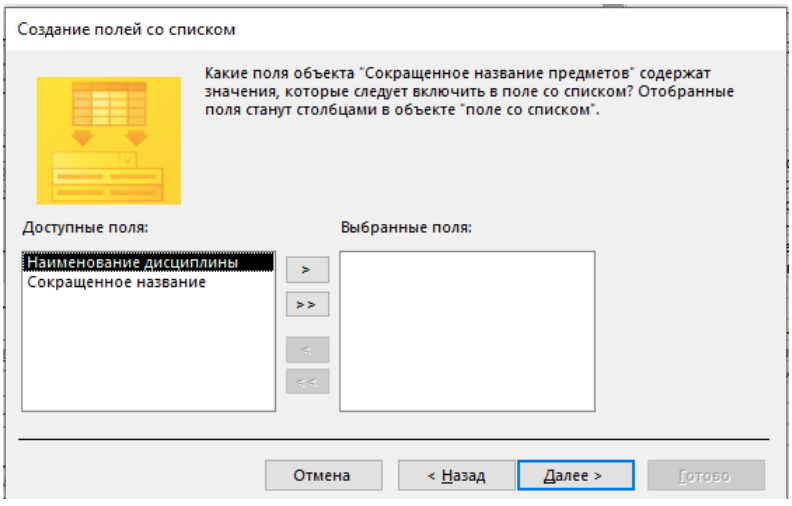
1. Вибір джерела. У цьому вікні необхідно вказати, звідки надходитимуть значення, що відображаються у списку. Нас цікавить пункт: *поле зі списком використовує значення таблиці або запиту* (він обраний за замовчуванням). Це задовольняє поставлене завдання, тому натискаємо Далі.



2. Зв'язок із таблицею чи запитом джерелом значень. Тут необхідно вибрати зі списку таблицю, на підставі полів якої формуватиметься список, що розкривається. За допомогою трьох перемикачів в блоці Показати, можна налаштувати список на відображення об'єктів, що цікавлять нас.

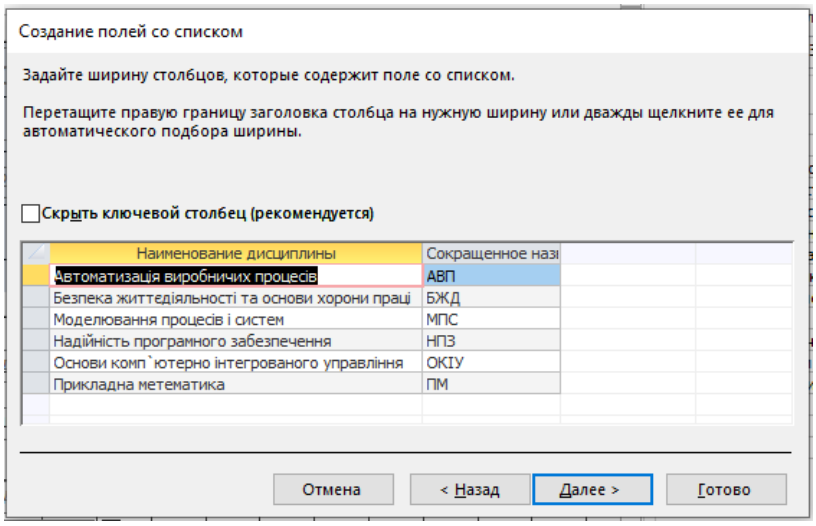


3. Вибір полів, що відображаються.

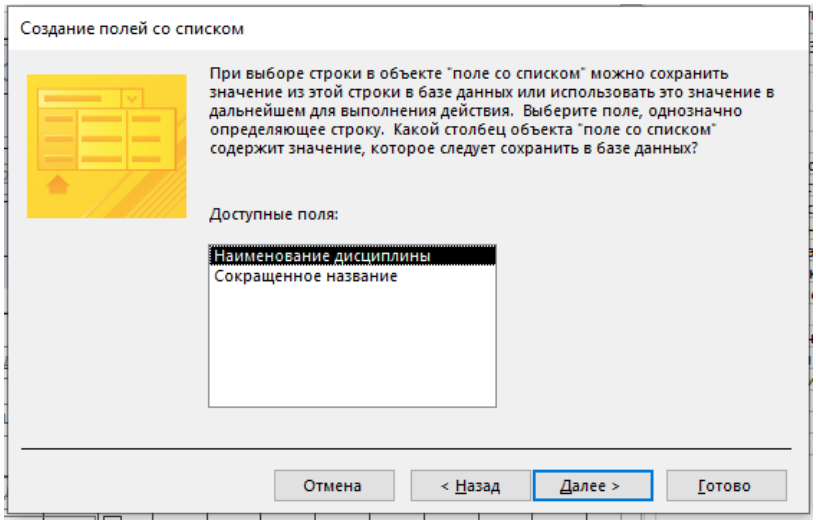


4. Вибір ширини стовпців. Якщо вибрано два і більше поля для відображення, тут можна відрегулювати їх ширину. Методи роботи такі, як у режимі конструктора. Подвійне клацання на межі поділу полів призводить до авто підбору ширини, за даними, що

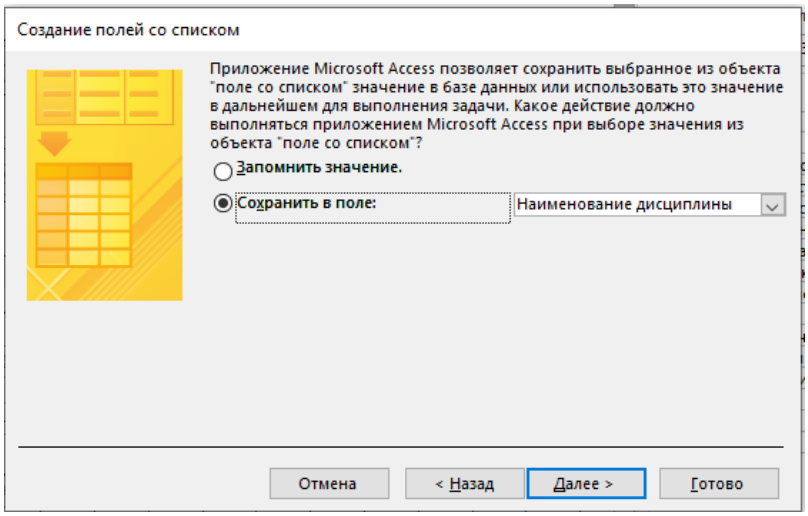
містяться в полі. Розташована у вікні галочка дозволяє приховати ключове поле. Це робиться, якщо інформація в ньому не повинна бути доступна користувачеві.



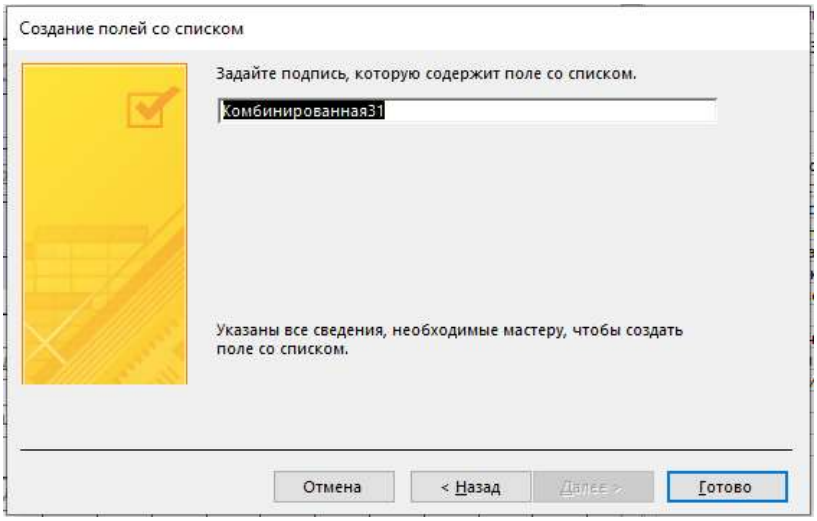
5. Вибір поля, яке буде збережено у базі даних.



6. Вказівка одержувача обраного значення. При виборі значення з списку, воно залишається в полі, після закриття повного списку.

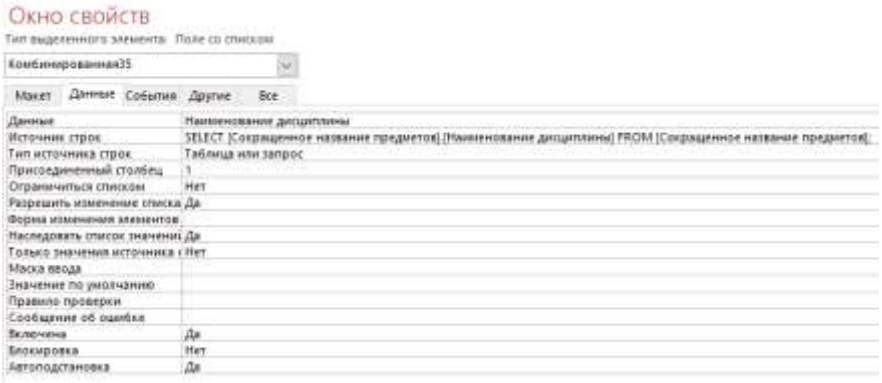


7. Створення підпису.



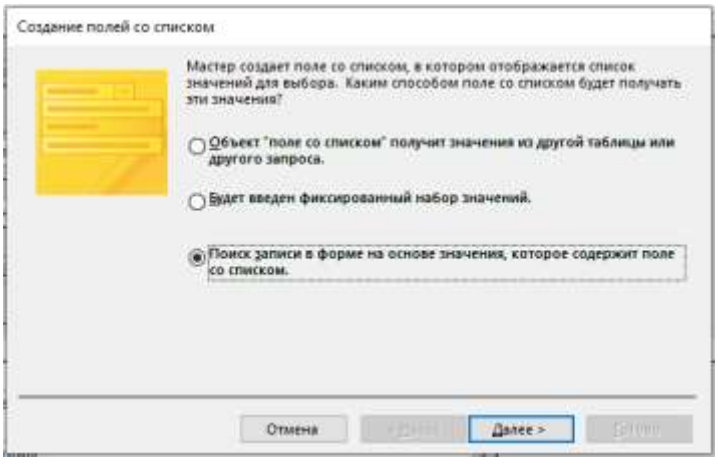
Елементи керування **Список** і **Поле зі списком** можна створити і без використання Майстра.

Для цього в режимі Конструктора додати у форму потрібний елемент (попередньо вимкнути кнопку Використання майстра). Потім внести потрібні коригування у вікно властивостей.



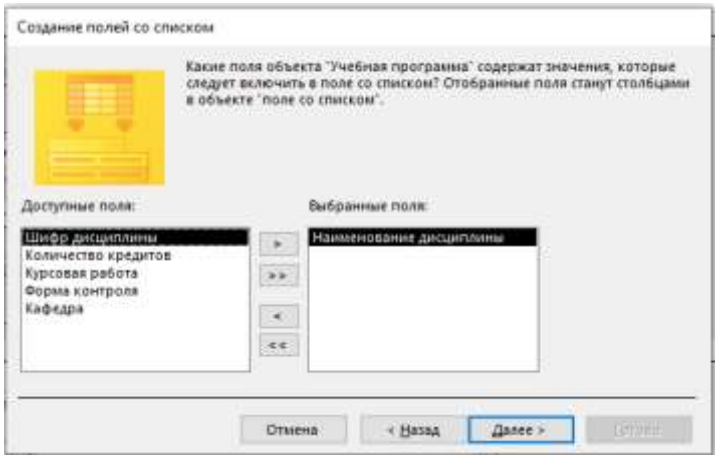
Пошук запису у формі на основі значення, яке містить поле зі списком.

Якщо під час створення поля зі списком було вибрано пункт: *Пошук запису у формі на основі значення, яке містить поле зі списком*, то послідовність дій буде такою:

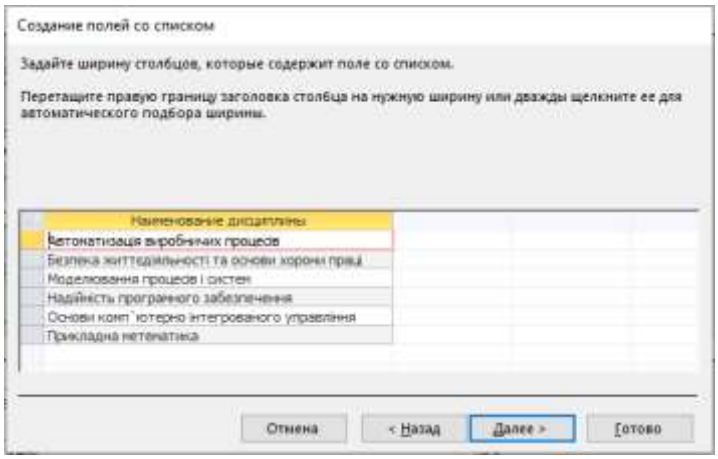
1. 

Як поле зі списком вибирає ключове поле:

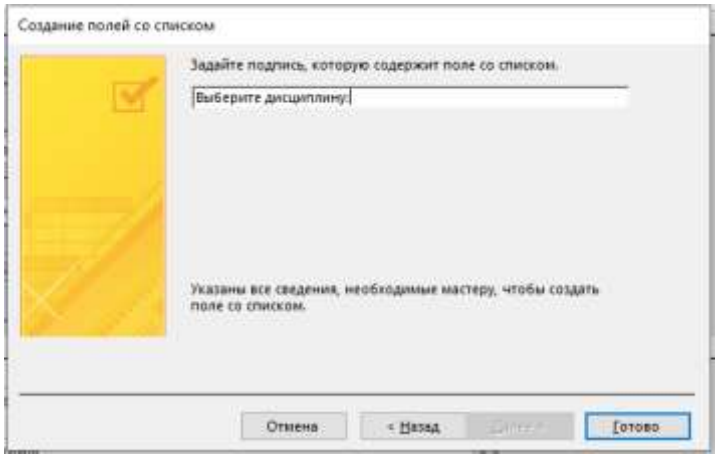
2.



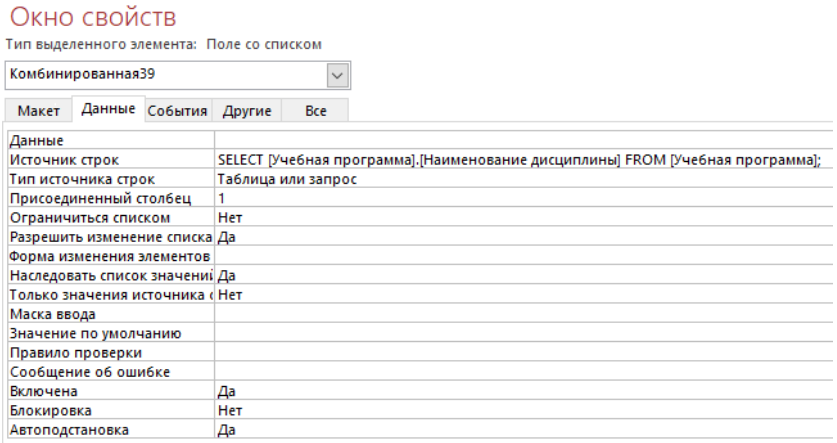
3.



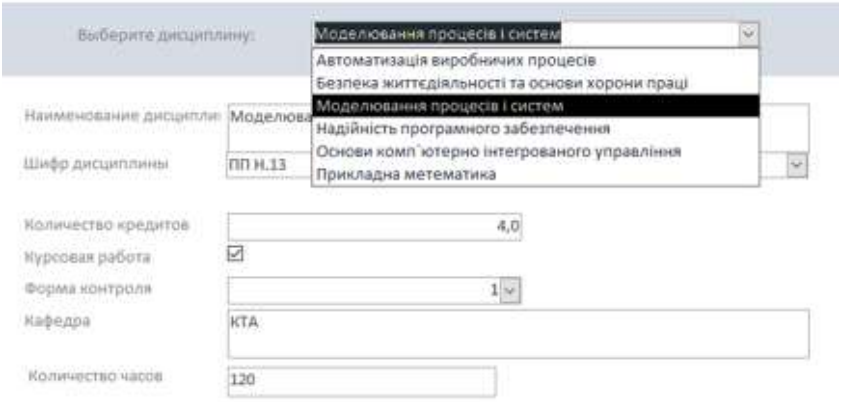
4.



Вікно властивостей у конструкторі матиме вигляд:



Елемент **Поле зі списком** матиме такий вигляд:



4.5.1.5 СТВОРЕННЯ КНОПКИ.

Кнопкиу формі використовуються для активізації дії чи послідовності дій. За допомогою кнопок можна, наприклад, організувати меню роботи програми. Щоб вказати, що має робити кнопка, необхідно зв'язати з нею макрос або процедуру обробки події, написану мовою VBA. Кнопки можна створювати за допомогою майстра та самостійно. Майстер Створення кнопок створює кнопки, які виконують стандартні дії. За допомогою таких кнопок можна відкрити або роздрукувати звіт або форму, переміститися до нового запису таблиці, видалити запис тощо. Майстер сам створює процедури обробки подій, що виконуються при натисканні кнопки. Щоб створити кнопку за допомогою майстра, потрібно:

1. Перебуваючи в режимі конструктора форми, переконайтеся, що увімкнено кнопку Майстра на панелі елементів.

2. Натиснути на кнопку Кнопка на панелі елементів,а потім на тому місці у формі, куди потрібно помістити кнопку. На екрані з'явиться вікно діалогу Створення кнопок.

3. Далі потрібно вибрати дію,яке має бути виконане при натисканні кнопки, та відповісти на питання майстра, пов'язані зі зробленим вибором.

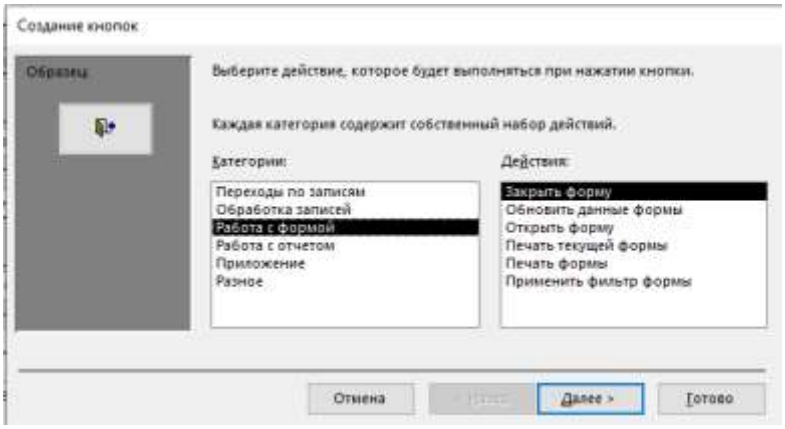
4. Введіть текст або виберіть потрібний малюнок для розміщення на кнопці.

5. В останньому вікні діалогу потрібно встановити зрозуміле ім'я кнопки і натиснути кнопку Готово.

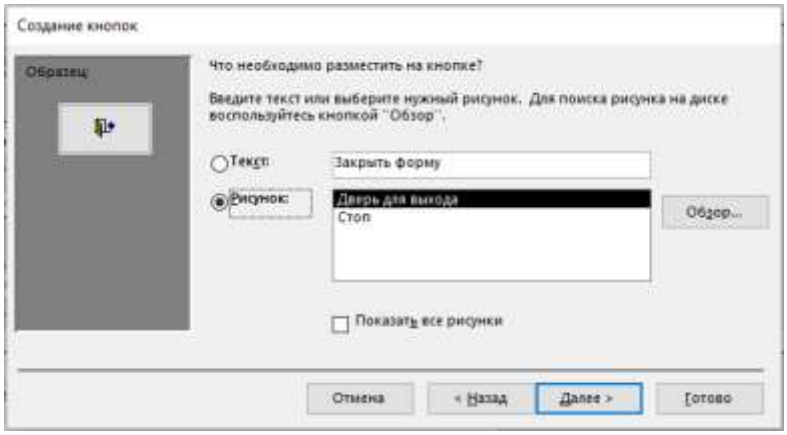
Перевірити дію кнопки можна, клацнувши по ній у режимі форми. Для перегляду та редагування процедури обробки події потрібно клацнути правою кнопкою миші на створеній кнопці та вибрати в контекстному меню пункт Обробка подій. Access зробить доступним текст процедури Ім'яКнопки_Click, яка запускається при натисканні на кнопку.

Приклад: Потрібно створитиу формі кнопку, натискання на яку дозволить запустити Excel.

Для цього клацанням по кнопці **Кнопка** на панелі елементів викличемо майстра **Створення кнопок**. Виберемо у лівому вікні **Категорії** пункт *Робота з формою*, а у правому вікні **Дії** зі списку можливих дій- Закрити форму.



Потім виберемо для розміщення на кнопці запропонований майстром малюнок або тексті натисніть кнопку**Готово**.



Accessстворить кнопку  , натискання на яку у режимі форми призведе до закриття форми. Під час встановлення покажчика миші на цю кнопку з'явиться підказка Закрити форму.

4.5.1.6 СТВОРЕННЯ ПІДЛЕГЛОЇ ФОРМИ, ЗВІТУ.

Елемент керування Підлегла форма/звіт призначений для розміщення в одній (головній) формі іншої (підлеглої) форми. Підлегла форма зазвичай застосовується для перегляду та редагування інформації у зв'язаних таблицях. У цій якості можна використати раніше створену форму.

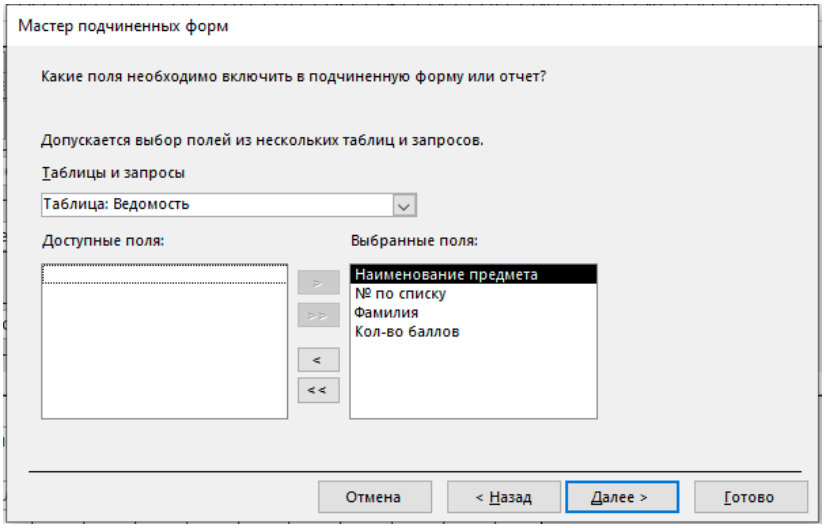
Для створення елемента керування Підлегла форма/звіт найпростіше скористатися послугами спеціального майстра. Для цього необхідно виконати такі дії:

1. Перебуваючи в режимі конструктора форми, переконатися, що кнопка Майстра включена на панелі елементів.

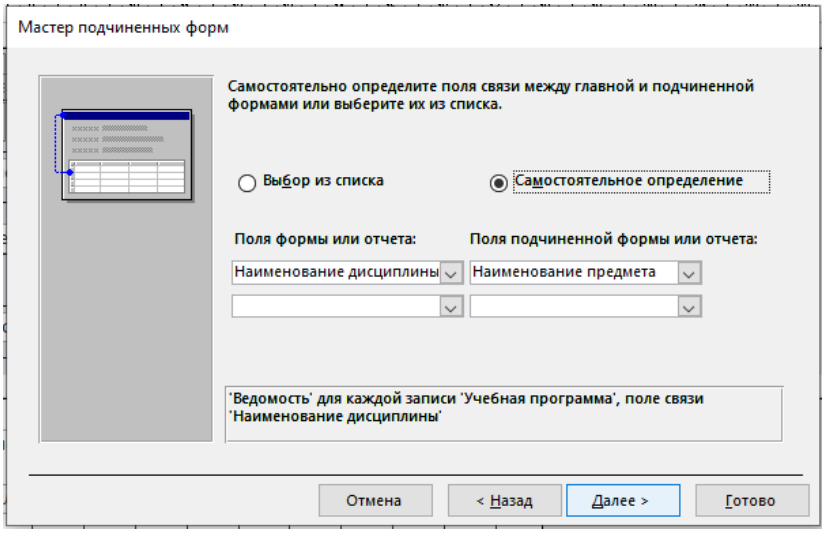
2. Потім натиснути кнопку Підлегла форма/звіт на панелі елементіві встановити покажчик миші те місце у формі, куди потрібно помістити підлеглу форму, і натиснути ліву кнопку. На екрані з'явиться перше вікно діалогу майстра Створення підлеглих форм та звітів.

3. Якщо вже існуєформа, яку можна використовувати як підлегла, слід вибрати значення перемикача форми, а потім вказати форму зі списку. Інакше потрібно вибрати значення таблиці або запиту.

4. Якщо готової підлеглої форми немає, то у другому вікні діалогу потрібно вибрати таблицю (запит), що є джерелом даних для підлеглої форми, та перекинути потрібні поля зі списку Доступні поля до списку Вибрані поля. Якщо передбачається встановлення зв'язку з головною формою, то список обраних полів повинні бути обов'язково включені поля, використовувані зв'язку.

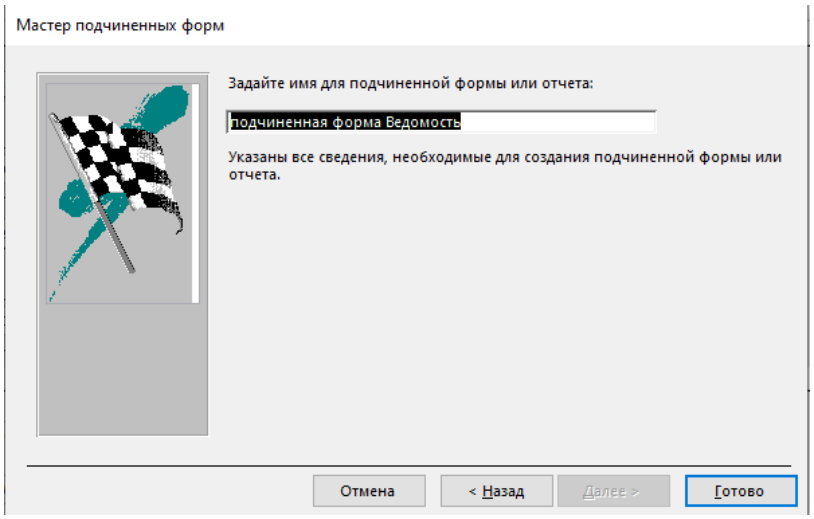


5. Далі майстер пропонує встановитизв'язок між головною та підлеглою формами. Є дві можливості: або скористатися списком можливих зв'язків, або самостійно вибрати поля зв'язку між головною та підлеглою формами.

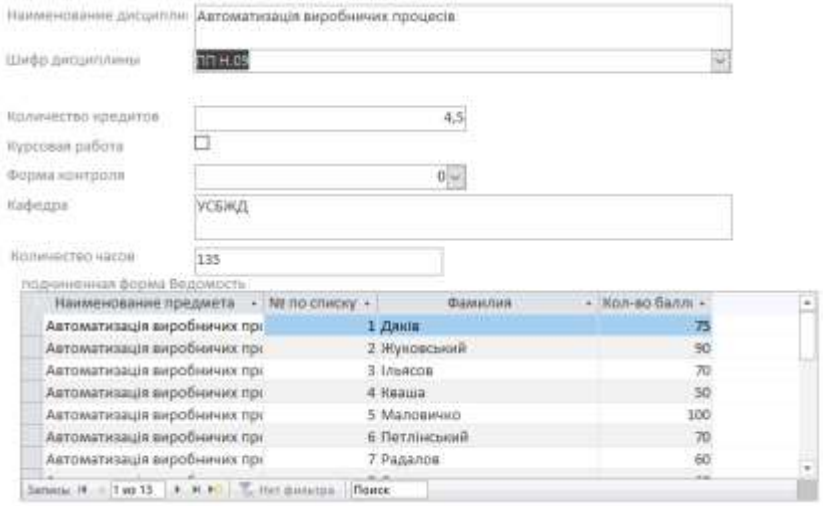


6. В останньому вікні діалогу визначається ім'я підлеглої форми, яка зберігається Access, як окрема форма.

Перевірити результат можна, перейшовши в режим форми.



7. Отриманний результат.



4.5.1.7 СТВОРЕННЯ НАБОРУ ВКЛАДОК.

За допомогою елемента керування Набір вкладок можна створювати багатосторінкові форми, подібні до вікна властивостей елемента керування. Цей елемент керування можна використовувати для економії місця на екрані та відображення інформації з кількох таблиць. На сторінках набору вкладок дозволяється розміщувати будь-які елементи керування, крім іншого набору вкладок.

Для створення цього елемента керування потрібно натиснути кнопку Набір вкладок на панелі елементівта клацнути лівою кнопкою миші в області даних форми для створення нового елемента керування. При відпустці кнопки миші Access створює елемент керування Набір вкладок, що складається з двох сторінок. Залежно від того, які дані та як Ви хочете відображати, може знадобитися включити до елемента керування Набір вкладок додаткові сторінки.

Для додавання сторінки до елемента керування Набір вкладок потрібно:

1. Клацнути правою кнопкою миші на цьому елементі керування. Ассесс виведе контекстне меню.

2. Вибрати пункт Додати вкладку. Access вставить нову сторінку за останньою сторінкою.

Для зміни порядку сторінок в елементі керування Набір вкладок потрібно:

1. Клацніть правою кнопкою миші на елемент керування Набір вкладок. Ассесс виведе контекстне меню.

2. Вибрати пункт Послідовність вкладок. Access виведе діалогове вікно Порядок сторінок із переліком сторінок.

3. У цьому списку потрібно вибрати сторінку, чию позицію ви хотіли б змінити та натискати кнопки Вгору або Вниз, доки сторінка не опиниться в потрібній позиції.

Для змініназви сторінки в елементі керування Набір вкладокпотрібно:

1. Клацнути правою кнопкою миші на корінці потрібної сторінки елемента керування Набір вкладок. Ассесс виведе контекстне меню.

2. Вибрати пункт Властивості. Access виведе діалогове вікно властивостей цієї сторінки.

3. У рядку Підпис введіть нову назву сторінки.

4.5.1.8 СТВОРЕННЯ СПЕЦІАЛЬНИХ ЕФЕКТІВ.

При розробці форм можна використовувати різні кольори, шрифти та інші спеціальні ефекти для надання форм привабливого вигляду. Прямокутники та лінії можна використовувати у формі для привернення уваги до важливої інформації, групування логічно пов'язаних елементів, проведення різних меж.

Щоб намалювати прямокутник, натисніть кнопку Прямокутник на панелі елементів, помістіть курсормиші в точку, де знаходиться верхній лівий кут прямокутника, і при натиснутій лівій кнопці миші перемістіть покажчик в нижній правий кут прямокутника. Якщо прямокутник закриває розміщені всередині нього елемента керування, у меню Формат виконайте команду На задній план. Щоб змінити розмір прямокутника, виділіть його та змініть за допомогою маркерів. Для переміщення прямокутника використається маркер переміщення.

Для малювання лінії натисніть кнопку Лініята проведіть потрібну лінію. Щоб створити строго горизонтальну або вертикальну лінію, потрібно під час її проведення утримувати клавішу Shift.

Рельєфне оформлення прямокутника або лінії можна здійснити за допомогою кнопок Колір лінії/кордону  , Товщина лінії/кордону, Звичайне оформлення  на панелі форматування.

4.5.1.9 РОЗМІЩЕННЯ МАЛЮНКІВ ТА ІНШИХ ОБ'ЄКТІВ OLE.

Для покращення видуформ та звітів Access у них можна розміщувати малюнки, діаграми та інші об'єкти, створені іншими програмами. Наприклад, у заголовок форми можна додати емблему фірми, а обсяги продажів у звіті подати у вигляді діаграми. Вставка

таких об'єктів у форми та звіти здійснюється за допомогою протоколу OLE (протоколу зв'язування та впровадження об'єктів).

Короткий опис протоколу OLE

Протокол OLE— це метод передачі у вигляді об'єктів (об'єктів OLE) між різними програмами Windows. Цей метод схожий на копіювання тексту та графіки в буфер обміну Windows з наступною вставкою в інші програми. Усі програми Microsoft Office підтримують протокол OLE.

Об'єкт OLE— це довільна частина даних, створених програмою Windows, що підтримує протокол OLE. Як об'єкт OLE може фігурувати як документ (наприклад, документ Word або електронна таблиця Excel), і його частина (фрагмент (тексту чи блок осередків електронної таблиці). Можна також використовувати різні графічні зображення (фотографії, малюнки, діаграми), відеокліпи, звукові файли та ін. Додаток, який використовується для створення об'єкта OLE, називається вихідним додатком, або програмою-сервером OLE, а файл, який містить цей об'єкт, - вихідним файлом. Об'єкт OLE крім самих даних також містить інформацію про вихідний файл.

Об'єкт OLE може бути пов'язаний або впроваджений у форму або звіт Access. Пов'язані та впроваджені об'єкти відрізняються місцезнаходженням даних та способом їх зміни після приміщення у форму.

При зв'язуванні створюється посилання об'єкт, що міститься у форму. Сам об'єкт залишається на своєму місці у вихідному файлі. Пов'язаний об'єкт автоматично оновлюватиметься під час оновлення вихідного файлу. Зв'язування об'єкта зручно застосовувати під час роботи з великими файлами, які небажано включати до файлу БД, а також з об'єктами, що використовуються у кількох формах та звітах. Якщо пов'язаний файл об'єкта переміщений, необхідно повторно встановити зв'язок.

При впровадженні об'єкта створюється його копія, яка вставляється у форму. Введений об'єкт стає частиною форми і втрачає зв'язок з вихідним файлом. При подвійному клацанні по впровадженому об'єкту він відкривається за допомогою програми-сервера, що створив його. Всі зміни, що вносяться в нього, відображаються в формі, що містить його.

Технологія OLE не лише забезпечує доступдо об'єктів, створених іншими програмами, а й суттєво спрощує процедуру їх зміни. Користувач може відкрити вихідну програму та внести зміни до об'єкта OLE, не припиняючи розробку форми або звіту.

Об'єкт OLEможе бути приєднаним чи вільним.

Приєднані об'єкти зберігаютьсяу файлі БД. Для зміни і навіть створення такого об'єкта не потрібно залишати Access. При зміні приєднаного об'єкта з Access змінюється лише об'єкт у БД, а вихідному файлі внесені зміни не відображаються.

Вільний об'єкт можна переглядатита змінювати, перебуваючи у формі чи звіті. Проте внесені до нього зміни зберігаються у вихідному файлі, а чи не в БД Access. Крім того, вихідний об'єкт може бути змінений без Access. При цьому внесені зміни будуть відображені у формі або звіті при їх подальшому відкритті.

Для відображення об'єктів OLE застосовуються два типи елемента керування: Приєднана рамка об'єкта та Вільна рамка об'єкта. Елемент Приєднана рамка об'єкта дозволяє відобразити у формі або звіті малюнки, діаграми та інші об'єкти OLE, що зберігаються в полях БД Access. Елемент Вільна рамка об'єкта використовується для відображення об'єктів, що зберігаються поза таблицями.

Використання елемента керування Приєднана рамка об'єкта

Приєднані рамки об'єктів слід використовувати для розміщенняу вигляді об'єктів OLE, які у полях таблиць. У режимі форми ці ЕУ використовуються для зображення, введення та зміни об'єктів у поточному записі таблиці точно так, як поля використовуються для зображення, введення та зміни тексту. У цьому кожен запис таблиці містить (чи містить) свій об'єкт.

Наприклад, поле Зображення таблиці Типи, включеноїв навчальну базу даних Борей містить малюнок для кожного включеного в цю таблицю типу товару. Для зображення цих малюнків у формі або звіті можна використовувати рамку об'єкта.

Щоб створити приєднанурамку об'єкта:

1. Відкрийте форму або звіт у режимі конструктора;

2. Виберіть Список поліву меню Вид (або натисніть кнопку Список полів на панелі інструментів);

3. Перенесіть поле, призначене для зберігання об'єктів OLE,макет форми або звіту. (Це поле має бути OLE-полем.).

Access створить рамку об'єкта, пов'язану із зазначеним полем. У режимі конструктора пов'язана рамка об'єкта зображається пустою; об'єкти із зазначеного поля зображуються в ній у режимі форми та попереднього перегляду або друкуються під час друку форми.

Користувачможе змінити розміри та пропорції об'єктів, розміщених у формі.

Щоб створити об'єктта впровадити його в приєднану рамку об'єкта:

1. Відкрийте форму в режимі форми (або відкрийте таблицю, форму або звіт у режимі таблиці) і знайдіть запис, до якого слід додати об'єкт.

2. Виділіть приєднану рамку об'єкта (або поле таблиці), до якої слід додати об'єкт.

3. Виберіть Вставити об'єкту меню Правка. На екрані з'явиться діалогове вікно Вставка об'єкта зі списком доступних програм-серверів OLE.

4. Виберіть Створити новий або Створити із файлуа потім виділіть тип об'єкта, який слід впровадити.

5. Якщоу формі повинен зображуватися не сам об'єкт, а значок, що його замінює, встановіть прапорець у вигляді значка.

6. Натисніть кнопку OK. Accessвідкриє вихідний додаток.

7. Створити об'єкт.

8. Для поверненняу Access виберіть Вихід у меню Файл вихідної програми, а потім ствердьте на пропозицію оновити документ.

Access впровадить створений об'єкту приєднану рамку об'єкта і відобразить або сам об'єкт (в режимі форми) або текст, що вказує тип об'єкта OLE, наприклад, Малюнок Paint (в режимі таблиці).

Використання елемента керування Вільна рамка об'єкту

Вільні рамки об'єктів слід використовувати для розміщенняу формах та звітах об'єктів OLE, які не потрібно заносити до таблиці (у такому разі рамка об'єкта не буде пов'язана з жодним полем таблиці). Наприклад, звіт Рахунок, включений до бази даних

Борей, містить емблему фірми. Ця емблема була розроблена професійним художникомза допомогою графічного редактора Paint, а потім збережена в окремому файлі. Щоб мати змогу зобразити у звіті іншу емблему або змінити існуючу, слід розмістити у макеті звіту вільну рамку об'єкта для емблеми.

Щоб створити вільнурамку об'єкта та помістити в неї існуючий об'єкт:

1. Відкрийте форму або звіт у режимі конструктора;

2. Натисніть кнопку Вільна рамка об'єкта на панелі елементів;

3. Встановіть вказівнику місце форми або звіту, куди слід помістити верхній лівий кут рамки об'єкта, і натисніть кнопку миші, щоб створити рамку стандартних розмірів, або вкажіть потрібні розміри рамки за допомогою миші. На екрані з'явиться вікно діалогу Вставка об'єкта зі списком OLE-програм, зареєстрованих у Windows;

4. Виберіть значення **З файлу**;

5. У полі Файл введіть повне ім'я файлу, який містить об'єкт, який потрібно ввести або зв'язати. Або натисніть кнопку Пошук та виберіть потрібний файл;

6. Якщо об'єкт слід зв'язати, а не впровадити, встановіть прапорець Зв'язок;

7. Якщоу формі повинен зображуватися не сам об'єкт, а значок, що його замінює, встановіть прапорець у вигляді значка;

8. Натисніть кнопку OK. Accessстворить вільну рамку об'єкта та зобразить у ній зазначений об'єкт.

Якщоу форму потрібно вставити малюнок, який надалі не вимагає змін, краще використовувати елемент керування Малюнок, тому що в цьому випадку форма завантажуватиметься швидше. Для цього слід клацнути по елементу керування Малюнок і вказати місце розташування графічного файлу, який містить малюнок, що вставляється. Також можна вибрати пункт меню Вставка, а потім Малюнок. Малюнок вибирається із зазначеного файлу та поміщається у рамку. Після цього можна змінити пропорції та розміри малюнка, але редагувати його не можна.

Вставлений малюнок вбудовується у форму і зберігає зв'язок з вихідним файлом. Проте якщо передбачається використовувати малюнок у кількох формах чи звітах, слід його пов'язати. Для зв'язування малюнка слід задати для властивості Тип малюнка елемента керування. Рисунок значення Пов'язаний. І тут малюнок буде зберігатися над БД, а вихідному файлі.

Малюнок можна увімкнутиу форму (звіт) і як фоновий малюнок, що займає її вікно. Якщо потрібно додати фоновий малюнок, слід використовувати властивість Малюнок форми або звіту.

4.6 НАЛАШТУВАННЯ ФОРМИ.

Налаштуванняформи проводиться шляхом зміни властивостей як самої форми, і її розміщених у ній елемент керування.

4.6.1 ЗМІНА ВЛАСТИВОСТЕЙ ЕЛЕМЕНТІВ КЕРУВАННЯ.

Властивості елемента керування, розміщеного у формі, викликаються через контекстне меню елемента керування або натисканням кнопки Властивості  на панелі

інструментів. Змінюючи деякі властивості елемента керування, можна налаштувати форму для зручнішого застосування.

Приєднаний елемент керування **Поле** успадковує за замовчуванням такі властивості поля таблиці або запиту, як формат поля, число десяткових знаків, маска введення, текст рядка стану, умова значення, повідомлення про помилку та інших. Ці властивості елемента керування можна змінити.

Для прискорення введення даних через форму можна для деяких елементів керування задати властивість Значення за замовчуванням, а у властивості Умова на значення задати перевірку значень, що вводяться. Щоб заборонити введення або зміну даних у режимі форми для приєднаного елемент керування, потрібно встановити для його властивості Блокування значення «Так». Для повної заборони змін, що виводяться у формі даних, простіше встановити властивості форми Дозволити зміни значення «Ні».

Іноді потрібно, щоб розміщений у формі елемент керування був невидимий у режимі форми. У цьому випадку слід встановити для його властивості Виведення на екран значення "Ні".

При розміщенні у формі кожен елемент керування отримує стандартне ім'я, наприклад «Поле3» або «Напис2». Це ім'я можна змінити більш змістовне, вказавши нове ім'я як Ім'я. Зазвичай це робиться, якщо передбачається використовувати ім'я даного елемента у будь-якому виразі чи програмі. Даючи елемент керування нове ім'я, потрібно стежити, щоб воно не співпало з ім'ям іншого елемент керування, розміщеного у формі.

Розміри елемент керування можна регулювати за допомогою маркерів розміру. Якщо елемент керування відображає дані текстового поля, що містить великий текст, то, крім збільшення розміру елемента керування, можна скористатися властивістю Смуги прокручування.

Властивість Розширення та Стиснення використовується під час друку форми. Якщо в цих властивостях встановлено значення "Так", то розміри елемента керування будуть при друку змінюватися автоматично так, щоб вміст був надрукований повністю.

Можна також змінити вираз для поля, що обчислюється (властивість **Дані**) та формат виведення на екран його значення (властивість **Формат поля**).

4.6.2 ЗМІНА ПОСЛІДОВНОСТІ ПЕРЕХОДУ.

Крім зміни властивостей елементів керування форми можна змінити послідовність переходу по полях форми. У режимі форми рух по полях форми здійснюється натисканням клавіші Tab або Shift+Tab (у зворотній послідовності). Послідовність цих переміщень задається в пункті Послідовність переходу меню Вид. У режимі конструктора форми при додаванні у форму нового елемента керування він додається і до Послідовності переходу. За допомогою команди Послідовність переходу з меню Вигляд може втрутитися в цей порядок і поміняти його на потрібну йому послідовність. У діалоговому вікні викликаної команди потрібно вказати на розташовану ліворуч від назви елемента керування кнопку, натиснути ліву кнопку миші та «перетягнути» елемент керування у потрібне місце у списку Послідовність.

4.6.3 ДОДАВАННЯ РОЗДІЛІВ.

Додавання або видалення розділіву форму здійснюється через меню Вид. У кожному розділі форми можна змінити, додати чи видалити будь-які елементі керування. Можна

змінювати розміри розділу. У розділах форми також є властивості, які можна міняти.

Так, розділ Заголовок форми можна відображати на екрані в режимі форми, якщо встановити значення Тільки на екран у властивості Режим виводу. У разі друку форми в цьому випадку розділ не друкується.

Можна заборонити відображення приміток на екраніу властивості Режим виводу, але дозволити їх друк або, навпаки, дозволити виведення приміток на екран, але не їхню роздруківку.

Розміри розділів у формі можна змінювати, переміщуючи їх нижню межу вгору або вниз лівою кнопкою миші.

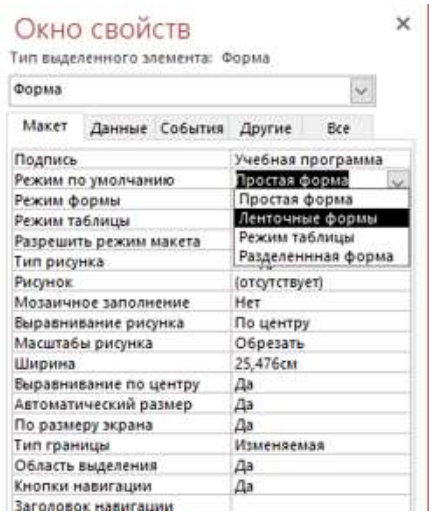
Властивості розділувикликаються подвійним клацанням миші на заголовку розділу або через контекстне меню, коли курсор миші знаходиться всередині розділу поза будь-якими елементами керування.

4.6.4 ЗМІНА ВЛАСТИВОСТЕЙ ФОРМИ.

Сама форма також має властивості, що викликаються подвійним клацанням миші з бланка форми, якщо покажчик знаходиться на сірій поверхні форми поза всіма розділами або на перетині горизонтальної та вертикальної лінійок.

У властивості задаються режими роботи з формою, її зовнішній вигляд, джерело даних та інші характеристики. Однією з основних властивостей форми є стандартний режим, який знаходиться на вкладці Макет. У цьому вся властивості встановлюється режим вікна форми під час її відкритті. Перерахуємо можливі режими:

- *проста форма*- на екрані відображається один запис;
- *стрічкова форма*- Відображення декількох записів;
- *таблиця*- Показ записів у табличному форматі.



У властивостях форми можна встановити порядок сортування її записів. Для цього слід вказати у властивості Порядок сортування імена полів, за якими проводиться сортування, розділивши їх комами. Якщо сортування проводиться по одному полю, то найпростіше задати його порядок у режимі форми або режимі таблиці, клацнувши мишею спочатку по цьому полю, а потім по одній з кнопок **Сортування за зростанням** або

Сортування за спаданням  , що задають потрібний напрямок сортування. Потрібний порядок сортування можна встановити безпосередньо у джерелі даних форми. Для цього потрібно клацнути за якістю Джерело записів, а потім по кнопці будівника запитів, що з'явилася праворуч. Відкриється вікно конструктора запитів, яке містить базову таблицю чи запит. Потрібно створити та зберегти запит, що задає потрібний вид сортування. Його інструкція SQL замінить як Джерело записів ім'я базової таблиці/запроса. Властивість Джерело записів можна використовувати і для іншої модифікації базового джерела даних форми, наприклад відбору підмножини записів відповідно до якого-або критерієм.

У властивостях форми визначається і режим роботи із записами. Режим може бути чотирьох типів, визначається значеннями Так/Ні:

- *Дозволити зміни* — можна переглядати, коригувати та вводити нові записи.
- *Дозволити додавання* — після переходу до останнього запису буде представлено порожній рядок для введення нового запису.
- *Дозволити видалення* — можна видаляти записи.
- *Ввід данних*— дозволяє розпочати роботу із записами з порожнього рядка, тобто з введення нового запису.

За промовчанням властивість Введення даних має значення "Ні". І тут при відкритті форми у ній виводяться існуючі записи. Якщо форма призначена для введення нових записів, необхідно встановити значення цієї властивості — «Так». Тоді при відкритті форми виводитиметься лише порожній запис.

5 ЗВІТИ.

5.1 ОСНОВНІ ПОНЯТТЯ.

Звіти- Форма подання інформації для використання та поширення. Звіти — підсумкові документи для осіб, яким було призначено створювану БД. Якщо форма - документ розробника та осіб, які працюють з інформацією в БД, то звіти - інструмент "господарів" БД, що дозволяє їм у потрібний момент мати інформацію для особистого використання або передачі іншим особам. Звіт необхідний підбиття підсумків діяльності у період підрахунків підсумкових сум тощо.

Структура звіту (рис. 5.1) нагадує структуру форми за одним істотним винятком — можливістю додавання кількох пар нових розділів, якщо виникає потреба угруповання даних за якими ознаками.

Заголовок звіту
Верхнійколонтитул
Область даних
Нижнійколонтитул
Примітка звіту

Рис. 5.1. Структура звіту

Можливість групування даних - це, мабуть, головна відмінність створення звіту від створення форми. Як реалізувати групування, ми докладно розглянемо нижче, а поки що коротко розглянемо саму ідею групування та її вплив на структуру розділів.

Припустимо, щоми будуємо звіт по таблиці, що описує реалізацію товарів різним клієнтам. Можна скласти звіт з різних «зрізів» такої таблиці. Наприклад, скласти звіт, що характеризує замовлення кожного клієнта. В цьому випадку основа групування даних- Клієнти. Усі замовлення розбиваються на групи, які стосуються одному клієнту.

Інший варіант- Розглянути, як реалізовувалися товари різних видів. І тут основа групування — окремий товар, проте дані про замовлення можна розбити на групи, які стосуються конкретного товару. Групи можуть бути вкладені. Наприклад, у першому випадку, всередині групування за клієнтами, можна згрупувати замовлення кожного клієнта за окремими товарами. Кожне групування супроводжується, як правило, появою двох розділів - заголовка групи та примітки групи. У заголовку зазвичай вказується інформація про поле - джерело групування (клієнти, товари) та ін. В області примітки можна підбити підсумки групування - кількість замовлень кожного клієнта, суму вартості всіх замовлень кожного клієнта, кількість замовлень кожного товару тощо.

Якщо групи вкладені, можна підбити проміжні підсумки, тобто.е. число замовлень конкретного товару, зроблених клієнтом. Загальна кількість вкладень груп — до 10. Властивості розділів звіту збігаються з властивостями розділів форми, хоча форм і звітів існують різні режими. Нагадаємо, що форма могла перебувати у трьох режимах: конструктора, роботи (форми) та попереднього перегляду.

У звіту також може бути три режими, причому два з них такі ж: режим попереднього перегляду та режим конструктора. Крім того, звіт може знаходитись у режимі зразка. Цей режим схожий на режим попереднього перегляду, але відрізняється від нього тим, що в режимі зразка відображається не весь звіт, а лише його частина, з метою оцінки того, як виглядатиме весь звіт. Режим, подібний до режиму форми, природно відсутній, оскільки зі звітом не працюють як з формою чи запитом.

Деякі розділи звіту, такі як колонтитулиі примітки мають додаткові властивості, більш тонкі, ніж у аналогічних розділів форм. Так, заголовок звіту може бути надрукований окремо на першій сторінці без колонтитулів.

Слід зазначити, що у звіті дублюються і багато інших властивостей різних об'єктів, що вивчалися нами у формах.

У звітах, як й у формах, розміщуються елементи управління: поля, написи тощо. буд. Як й у формах, може бути пов'язаними, вільними і обчислюваними залежно від джерела інформації, реалізованої елементі управління.

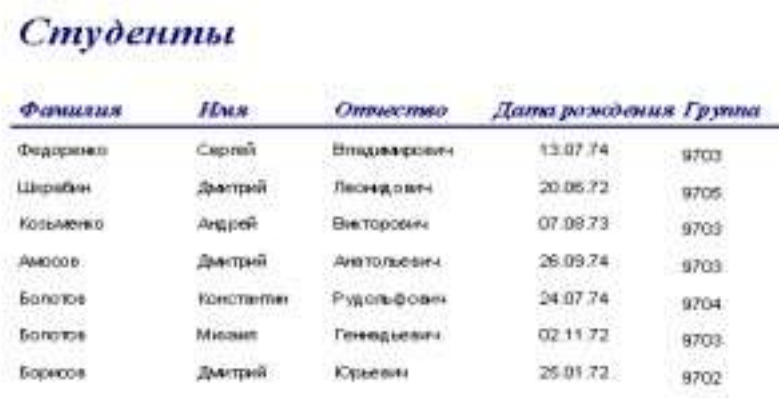
5.2 СТВОРЕННЯ ЗВІТУ.

Звіти, як і форми, можна створювати за допомогою майстрів чи самостійно. Джерелом даних для звіту також є таблиці та запити. Якщо у звіті потрібно надати дані з різних таблиць, має сенс попередньо створити багатотабличний запит, а потім будувати звіт на його основі.

Щоб створити звіт, потрібноу вікні БД навести курсор на цікаву для нас таблицю або запит і на вкладиші Створення вибрати піктограму Звіт. На основі вибраного об'єкта буде автоматично сформовано стрічковий звіт.

У цих звітах є заголовок, що містить ім'я звіту, що збігається з назвою таблиці або

запиту, і дата створення звіту. Нижній колонтитул містить номер сторінки. В області даних містяться всі поля таблиці (запиту), розташовані в один стовпець або в табличному вигляді (у стрічковому звіті). На одній сторінці реалізується кілька записів таблиці (запиту). Приклад стрічкового звіту наведено на рис. 5.2.



Студенты

Фамилия	Имя	Отчество	Дата рождения	Группа
Федоренко	Сергей	Владимирович	13.07.74	9703
Шарабин	Дмитрий	Леонидович	20.05.72	9705
Козьменко	Андрей	Викторович	07.08.73	9703
Амосов	Дмитрий	Анатольевич	26.09.74	9703
Болотов	Константин	Рудольфович	24.07.74	9704
Болотов	Михаил	Геннадьевич	02.11.72	9703
Борисов	Дмитрий	Юрьевич	25.01.72	9702

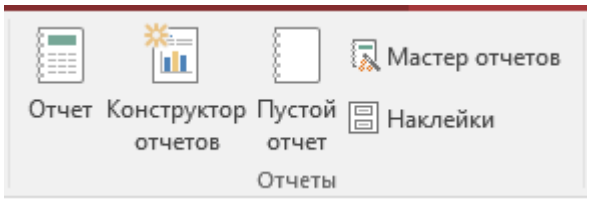
Рис. 5.2. Приклад стрічкового звіту

5.2.1 СТВОРЕННЯ ЗВІТУ ЗА ДОПОМОГОЮ МАЙСТРА ЗВІТІВ.

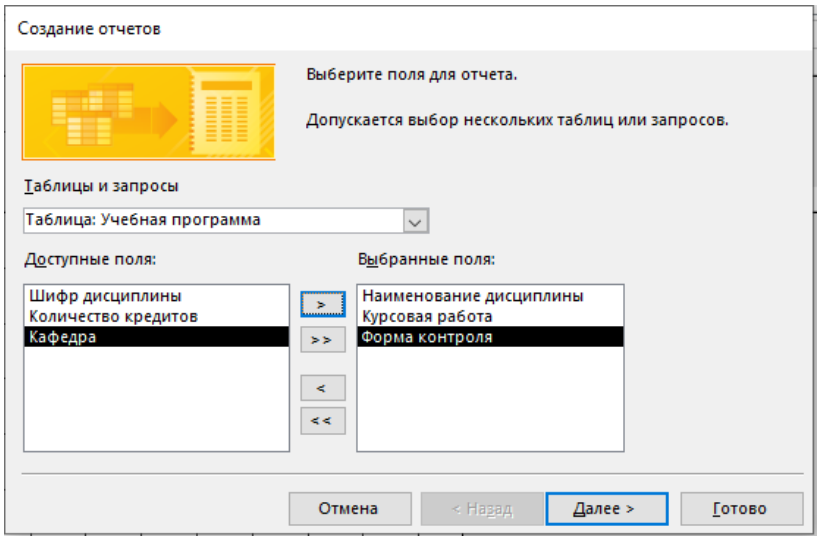
Використання майстра з розробки звітівє швидким та захищеним від помилок способом створення нового звіту. Звіт, створений майстром, можна використовувати у вигляді, у якому він було створено, чи поліпшити шляхом внесення змін.

Щоб створити звітза допомогою майстра з розробки звітів треба:

1. Вибрати спосіб створення звіту **Майстер звітів**.

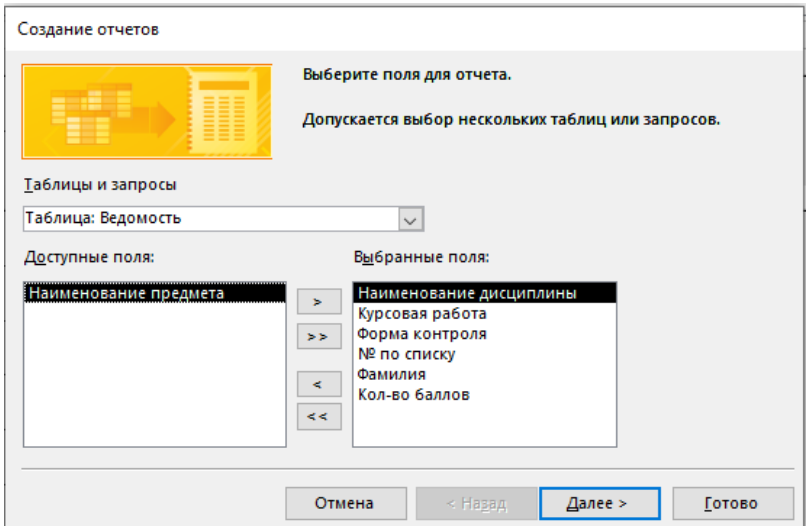


2. На екрані з'явиться перше вікно **звіту**.

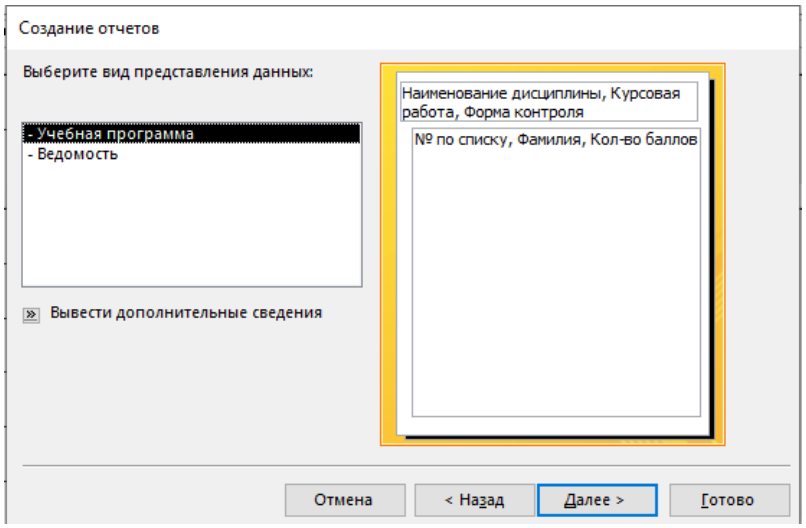


3. Зі списку вибрати таблицю/запит, що містить дані, які слід подати у звіті.

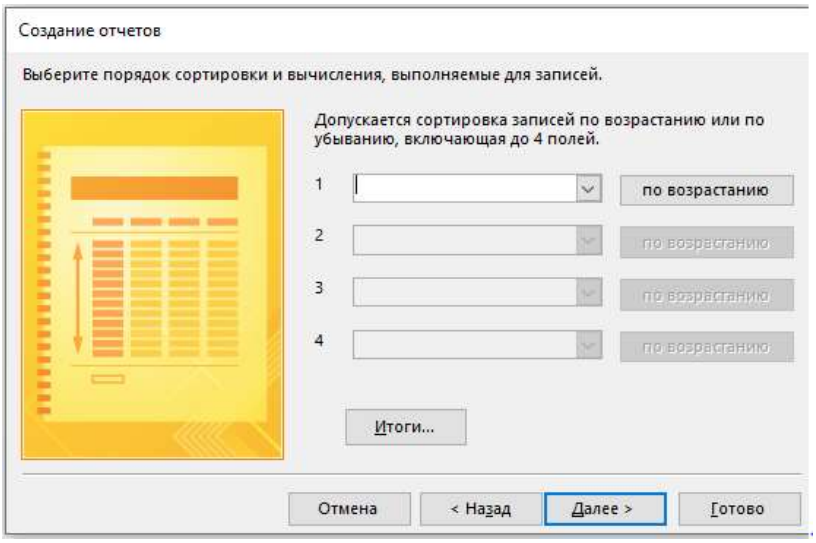
4. Переслати необхідні для звіту поля зі списку Доступних поліводо списку Вибраних.



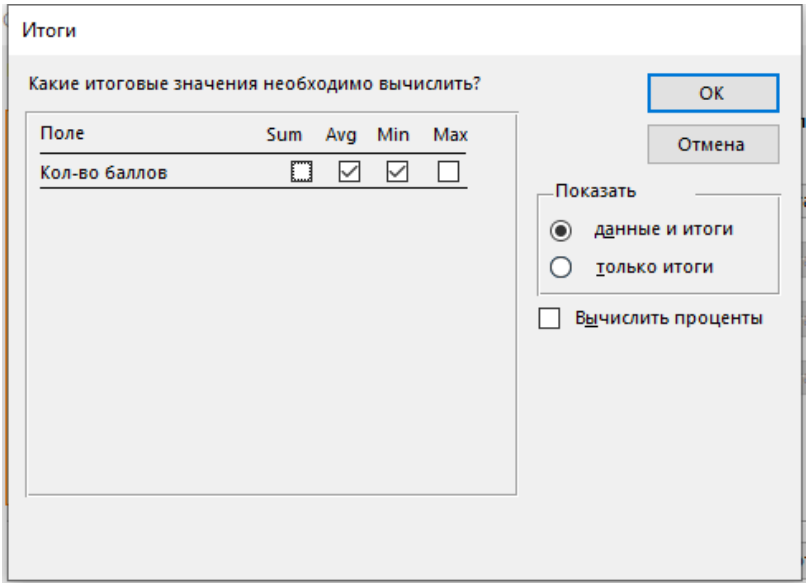
5. У наступному вікні встановити необхідні рівні угруповання.



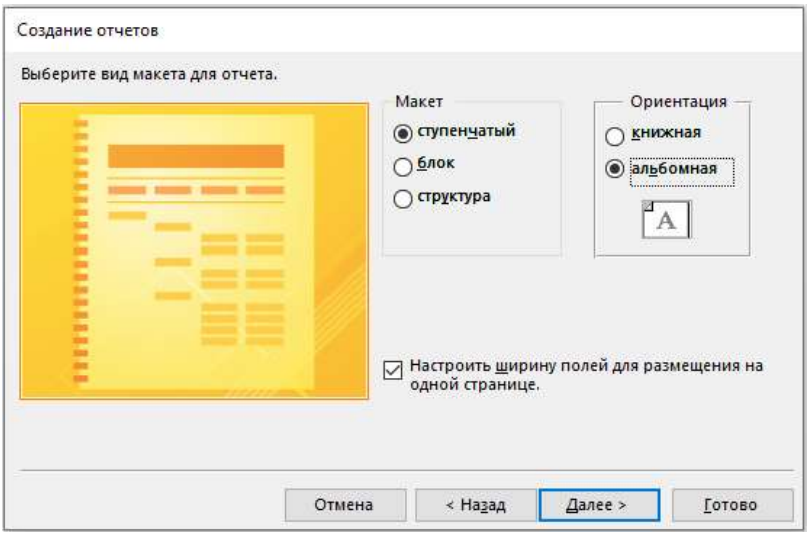
6. Даніу звіті можна відсортувати. Можна встановити сортування до чотирьох рівнів, за спаданням або зростанням.



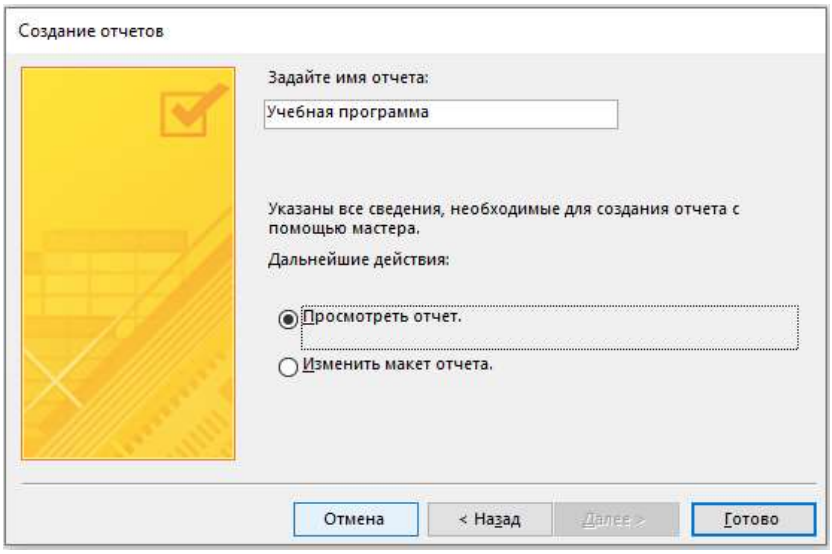
7. Натиснувши кнопку Підсумки…, перейдемо до вікна, яке дозволить вибрати для звіту необхідні підсумкові значення.



8. На наступному етапі задається макет звіту.



9. В останній вікні потрібно задати ім'я звіту.



Створіннязвіту майстром закінчено. Ви можете переглянути зразок звіту. Для цього можна натиснути кнопку Попередній перегляд. Якщо звіт потрібно дещо змінити, скористайтесь Конструктором.

5.2.2 СТВОРЕННЯ ЗВІТУ ЗА ДОПОМОГОЮ КОНСТРУКТОРА.

Користувачможе почати з порожнього звіту та самостійно розмістити у ньому всі необхідні поля, написи та інші елементи керування.

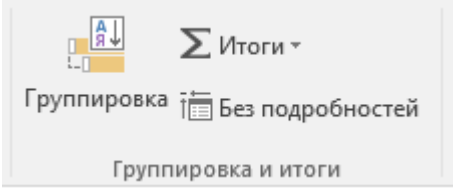
Для створення звіту бездопомоги майстра потрібно, знаходячись у вікні БД на вкладці Створення вибрати Порожній звіт.

На екрані з'явиться порожній звіт у режимі макету. З правого боку з'явиться меню, яке дозволить вибрати із потрібних таблиць поля, на основі яких буде сформовано звіт.

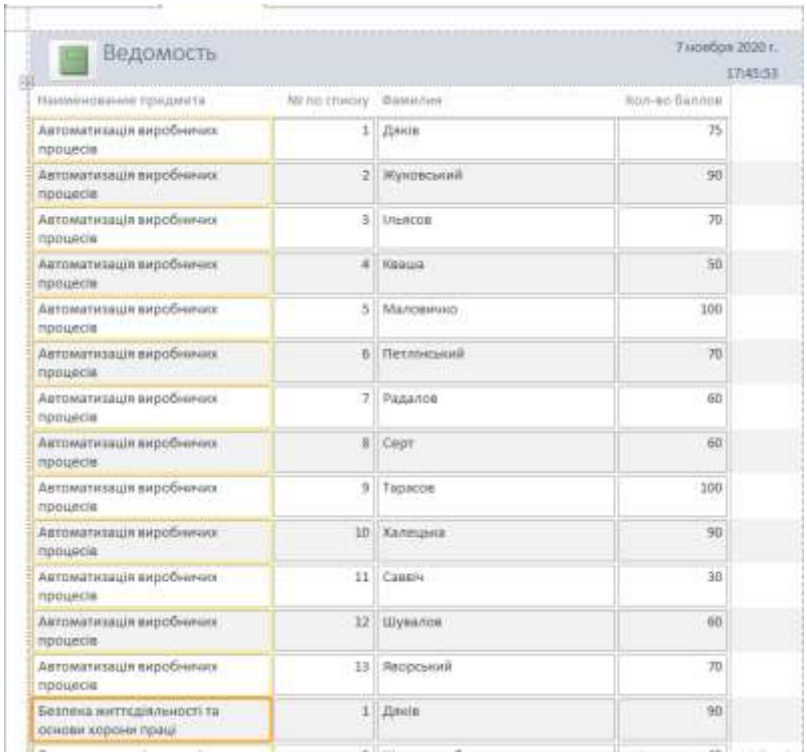
Усі елементи керування створюються розділах звіту тими самими методами, як у формах. Зупинимося на принциповій відмінності звіту від форми створення груп.

5.3 УГРУПОВАННЯ ТА ПІДСУМКИ.

Зазвичай записиу звіті потрібно розмістити у визначеному порядку. У меню Конструктор є підкоманда Угруповання та підсумки.



Приклад: На основі таблиці Відомість створимо простий звіт.



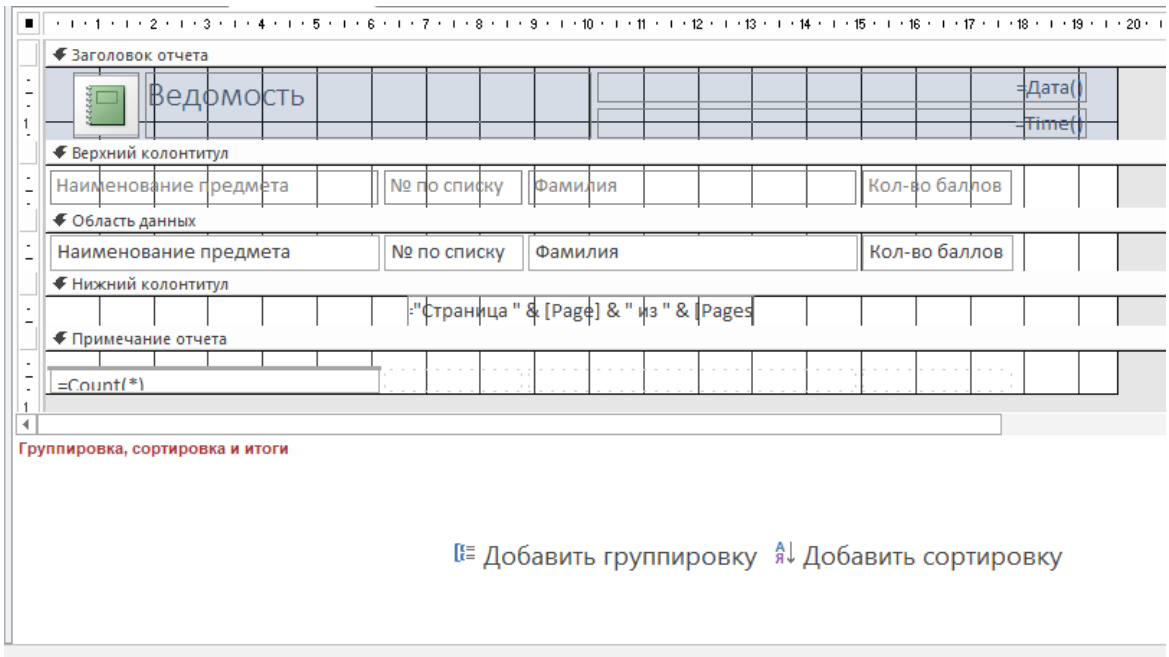
Ведомость

7 ноября 2020 г.
17:45:53

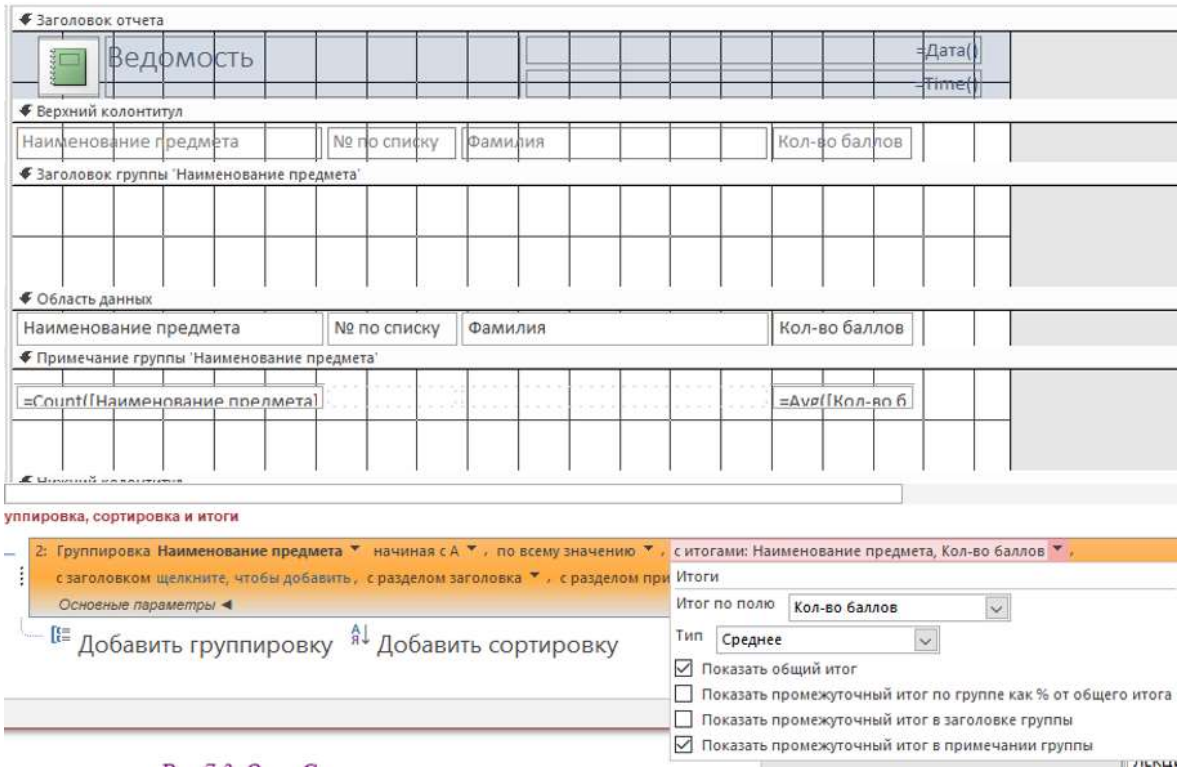
Наименование предмета	№ по списку	Фамилия	Кол-во баллов
Автоматизація виробничих процесів	1	Дяків	75
Автоматизація виробничих процесів	2	Жуковський	90
Автоматизація виробничих процесів	3	Ілляєв	70
Автоматизація виробничих процесів	4	Кваша	50
Автоматизація виробничих процесів	5	Маловичко	100
Автоматизація виробничих процесів	6	Петлюнський	70
Автоматизація виробничих процесів	7	Радалов	60
Автоматизація виробничих процесів	8	Серт	60
Автоматизація виробничих процесів	9	Тарасов	100
Автоматизація виробничих процесів	10	Халецька	90
Автоматизація виробничих процесів	11	Савіч	30
Автоматизація виробничих процесів	12	Шувалов	60
Автоматизація виробничих процесів	13	Яворський	70
Безпека життєдіяльності та основи охорони праці	1	Дяків	90

Перейдемо у режим конструктора.

При натисканні на угруповання, внизу вікна конструктора з'являться наступні команди:

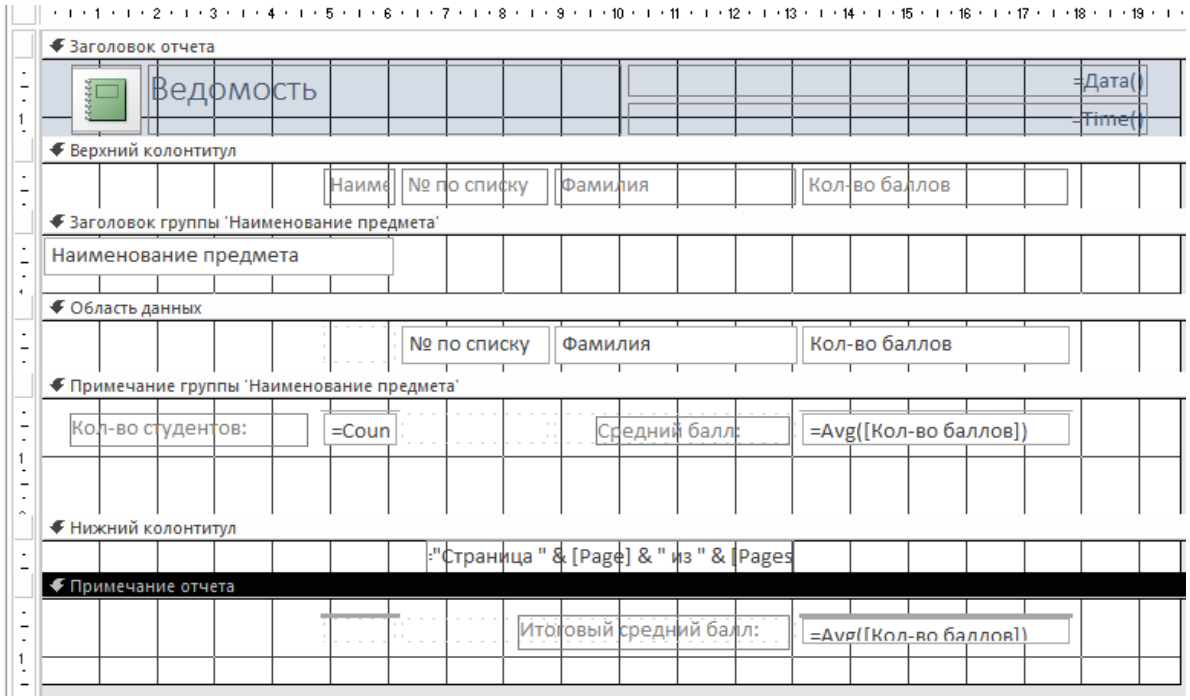


За допомогою цих команд можна просто провести сортування звітних даних або визначити групування даних.

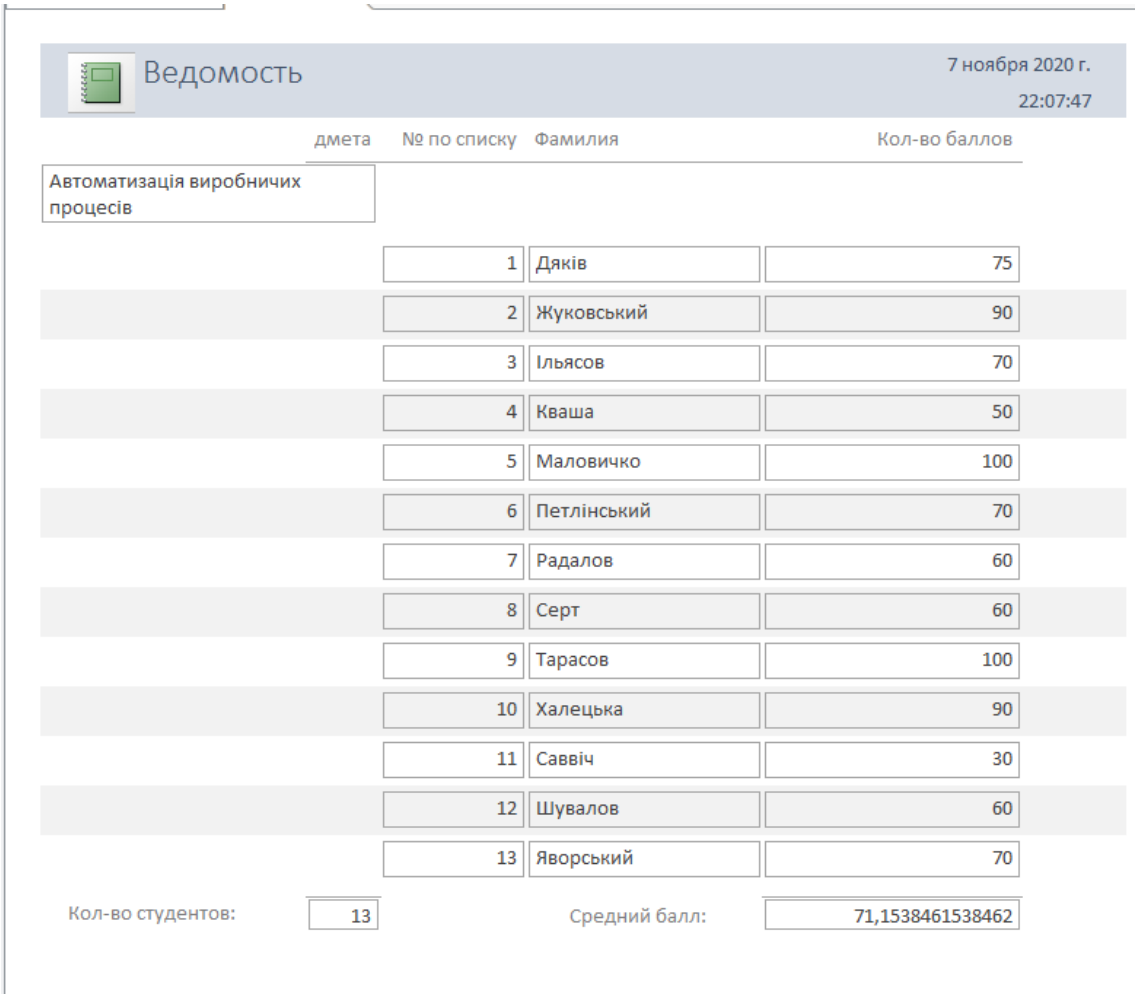


Потім додамо сортування за полем № за списком.

Для того, щоб поле Найменування предмета з'являлося один раз перед групою записів по студентам, які здавали цей предмет, перенесемо його з області даних у заголовок групи:



В результаті отримаємо звіт наступного виду, де відображені і проміжні підсумки групи записів щодо кожного предмета.



Ведомость

7 ноября 2020 г.
22:07:47

дмета	№ по списку	Фамилия	Кол-во баллов
Автоматизація виробничих процесів			
	1	Дяків	75
	2	Жуковський	90
	3	Ільясов	70
	4	Кваша	50
	5	Маловичко	100
	6	Петлінський	70
	7	Радалов	60
	8	Серт	60
	9	Тарасов	100
	10	Халецька	90
	11	Саввіч	30
	12	Шувалов	60
	13	Яворський	70

Кол-во студентов: 13 Средний балл: 71,1538461538462

5.4 РЕЖИМИ ВІКНА ЗВІТУ.

Вікно звітуможе знаходитися в одному з трьох режимів: режимі конструктора, режимі перегляду зразка та режимі попереднього перегляду:

Режим конструктора призначений для створення новихта зміни існуючих звітів;

Режим перегляду зразка призначений для попередньої оцінки правильності шрифтового оформлення та розташування елементів керування у звіті. У цьому режимі відображаються всі розділи звіту, а також кілька записів даних. При цьому виконується сортування та групування даних, однак не використовуються умови відбору, ні об'єднання таблиць, визначені в базовому запиті.

У режимі попереднього перегляду звіт виглядає на екрані так, як він виглядав би надрукованим. Надрукований звіт може виглядати інакше, ніж на екрані в режимі попереднього перегляду, якщо для його оформлення використовуються шрифти, що не масштабуються. Масштабуються шрифти Tгуе Туре та деякі інші шрифти.

Під час перегляду звіту в режимі перегляду зразка або попереднього перегляду можна змінювати масштаб зображення звіту.

5.5 ДРУК ЗВІТУ.

Щоб надрукувати готовий звіт, потрібно виконати команду Друк із меню Файл.

Надрукований звітможе містити пусті сторінки. Як правило, це є наслідком того, що розмір звіту перевищує розмір паперу, який використовується для друку.

Щобу звіті не було порожніх сторінок, сума повної ширини звіту з шириною правого та лівого полів не повинна перевищувати ширину паперу, вказану при налаштуванні друку (дивіться таку формулу):

Ширина звіту + Ліве поле + Праве поле <= Ширина паперу

Якщо звіт містить надто багато порожнього простору навколо розділівта елементів управління, то його можна зменшити, змінивши розміри та визначивши деякі властивості розділів та елементів керування. Під час друку звіту дуже важливо правильно встановити властивості звіту, його розділів та деяких елементів керування. Властивості звіту викликаються подвійним клацанням миші прямокутником на перетині горизонтальної та вертикальної лінійок, або з контекстного меню, викликаного, коли покажчик миші знаходився в сірій області бланка звіту поза розділами. Властивості розділів викликаються подвійним клацанням миші по сірій смузі під назвою розділу. Властивості будь-якого елемента керування викликаються подвійним клацанням миші з цього елемента керування.

Для друку всього звіту важливі властивості: Верхній колонтитул і Нижній колонтитул.

Можливі значення:

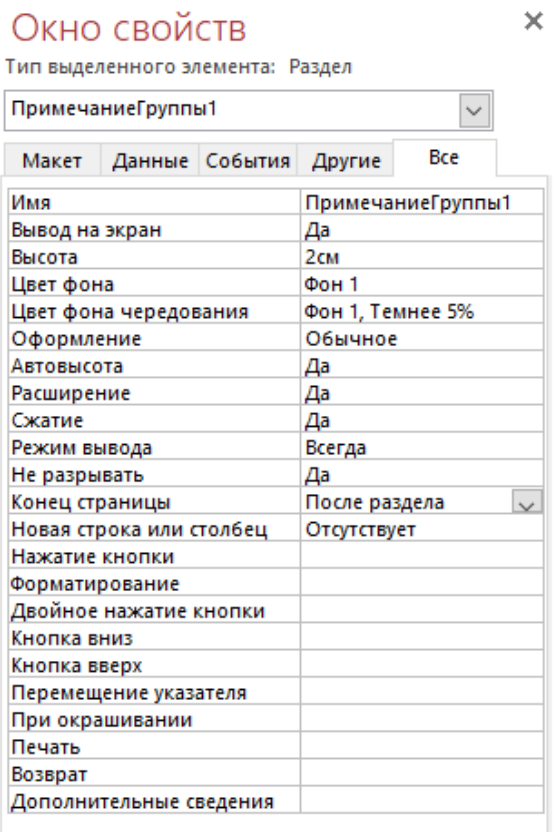
- *Усі сторінки* — колонтитули друкуються на всіх сторінках;
- *Без заголовка* — колонтитули не друкуються на сторінках, де друкувався заголовок звіту;
- *Без примітки*— колонтитули не друкуються на сторінках, де друкувалося примітка звіту;
- *Без заголовка/примітки*— колонтитули не друкуються на сторінках, де друкувалися заголовок або примітка звіту.

Для друку розділів звіту важливі властивості Кінець сторінкиі Не розривати.

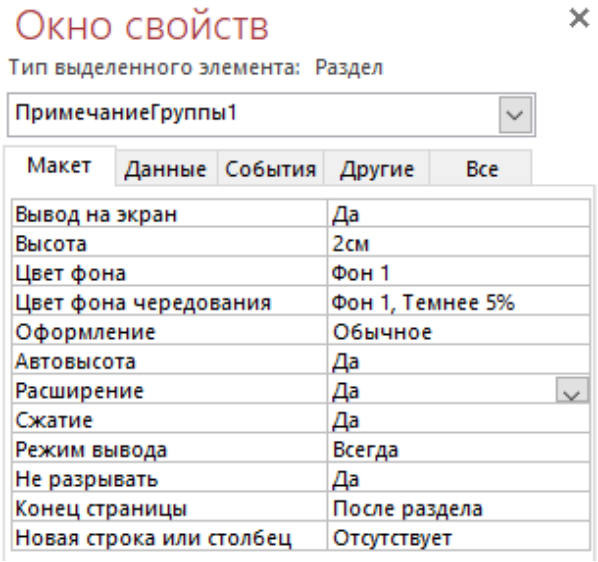
Властивість Кінець сторінкиможе приймати значення:

- *Відсутня* — розставляти сторінки, не звертаючи уваги на початок/кінець розділу;
- *До розділу* — поставити кінець сторінки перед початком розділу;
- *Після розділу* — поставити кінець сторінки після кінця розділу;
- *До та після розділу* — поставити кінець сторінки до початку та після кінця розділу.

Для того щобу звіті кожна група друкувалася на окремій сторінці, потрібно встановити значення Після розділу у властивості Кінец сторінки розділу Примітка групи Група. Щоб розділи не розривалися під час друку, потрібно задавати значення «Так» у властивості Не розривати.ь



Під час друку звіту важливо виділити місце для елементів керування , особливо якщо це довге символьне поле або поле типу Мемо. У елементів керування типу поле є властивості **Розширення** та **Стиснення**. Якщо задати значення «Так» у властивості Розширення, при необхідності автоматично збільшуватиметься висота області, виділеної полю на бланку звіту. Розділ, в якому знаходиться такий елемент керування, автоматично отримує значення "Так" у властивості Розширення. За промовчанням замість полів, що містять порожні значення, на сторінках звіту залишається порожній простір. Для видалення цього порожнього простору слід використовувати властивість Стиснення. Поле, що має значення "Так", у цій властивості не займає місця під час друку звіту, якщо містить порожнє значення або рядок нульової довжини.



6 МАКРОСИ.

Макрос - програма, що складається з послідовності макрокоманд (макрос від слова "макрокоманда").

Макрос може бути поряд з іншими об'єктами представлений як окремий об'єкт (**ізольований макрос**), який відображається в області навігації групи Макроси . Крім того, макрос, пов'язаний з будь-якою подією у формі, звіті або елементі управління, може бути впроваджений у форму або звіт (**впроваджений макрос**). При цьому він не відображається як об'єкт групи Макроси , а стає компонентом форми або звіту.

Кожен макрос складається з однієї чи кількох макрокоманд. Залежно від контексту, в якому ви працюєте, деякі макрокоманди можуть бути недоступними.

Макрокоманда— це інструкція, орієнтована виконання певного дії над об'єктами Access та його елементами.

Наприклад, макрокомандою можна відкрити форму, звіт, надрукувати звіт, запустити виконання запит, застосувати фільтр, присвоїти значення, створити своє меню, організувати виконання різних гілок алгоритму залежно від умов. Макрокоманда Запуск Команди Меню дозволяє виконати будь-які вбудовані команди Access, які виводяться на вкладках стрічки або в контекстних меню та відповідають режиму виконання макрокоманди. Наявний в Access набір макрокоманд реалізує практично будь-які дії, які необхідні для вирішення задачі.

Макрос може бути поряд з іншими об'єктами представлений як окремий об'єкт (ізольований макрос), який відображається в області навігації групи Макроси . Крім того, макрос, пов'язаний з будь-якою подією у формі, звіті або елементі управління, може бути впроваджений у форму або звіт (впроваджений макрос). При цьому він не відображається як об'єкт групи Макроси, а стає компонентом форми або звіту.

Макроси можуть запускатися виконання прямо з області навігації. Можливе вирішення завдань за допомогою низки взаємопов'язаних макросів, перший з яких запускатиметься на виконання в області навігації. Користувач запускає головний макрос виконання і далі все управління виконанням завдання здійснюється зсередини макросу. Макрос сам відкриває потрібні об'єкти, вибирає та обробляє дані, викликає інші макроси, слідуючи алгоритму, що призводить до вирішення задачі. При необхідності з макросу може бути ініційований діалог із користувачем. Для переходу з різних гілок макросу використовується блок управління **Якщо** (If).

Ізольований макрос може виконуватися у відповідь на численні види подій, що виникають у формах, звітах та їх елементах управління. Впроваджений макрос завжди пов'язується з подією та зберігається у формі чи звіті. Події настають, насамперед, під час виконання певних дій користувача з об'єктами.

Прикладами подій є:

- зміна даних у полі,

- відкриття або закриття форми чи звіту,

- натискання кнопки у формі та просто передача фокуса від одного поля до іншого.

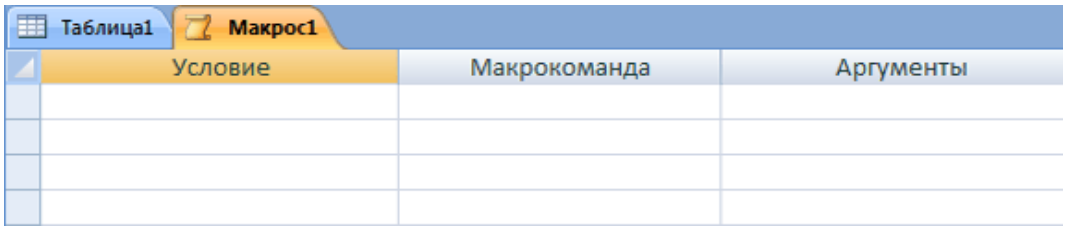
Зв'язок макросів з подіями дозволяє автоматизувати програми, використовуючи макроси для відкриття форм, друку звітів, виконання послідовності запитів, для виконання дій, що залежать від значень деякого поля в базі даних, для виведення повідомлень користувача або відключення попереджувальних повідомлень під час виконання запитів дії

і багато іншого . Збереження впроваджених макросів разом із формами та звітами спрощує управління об'єктами докладання.

6.1 КОНСТРУЮВАННЯ МАКРОСУ.

У Access 2010 з'явився новий конструктор макросів, що спрощує створення складних макросів і дозволяє скоротити кількість помилок під час кодування. Для цього в конструкторі застосовуються списки, що розкриваються, технологія IntelliSense, повторне використання існуючих макросів, перетягування, а також копіювання і вставка через буфер обміну.

У попередніх версіях Microsoft Access конструктор макросів складається з трьох стовпців. Умовні оператори додаються в стовпець Умова, макрокоманди - в стовпець Макрокоманда, а вказані параметри - в стовпець Аргументи.



Новий конструктор макросів для Access 2010 більше нагадує текстовий редактор. Трьох стовпців більше немає. Замість них макрокоманди і умовні оператори відображаються в списках, що розкриваються, у звичному для програмістів форматі. Аргументи відображаються у вбудованому діалоговому вікні.

Конструктор надає кілька способів додати дію у макрос. Найпростіший - скористатися списком, що випадає, або двічі клацнути її в каталозі команд. При додаванні команди до будівельника макросів з'являються додаткові параметри. Наприклад, при додаванні команди If (якщо) стають доступними параметри, що дозволяють створювати складні вкладені умови.

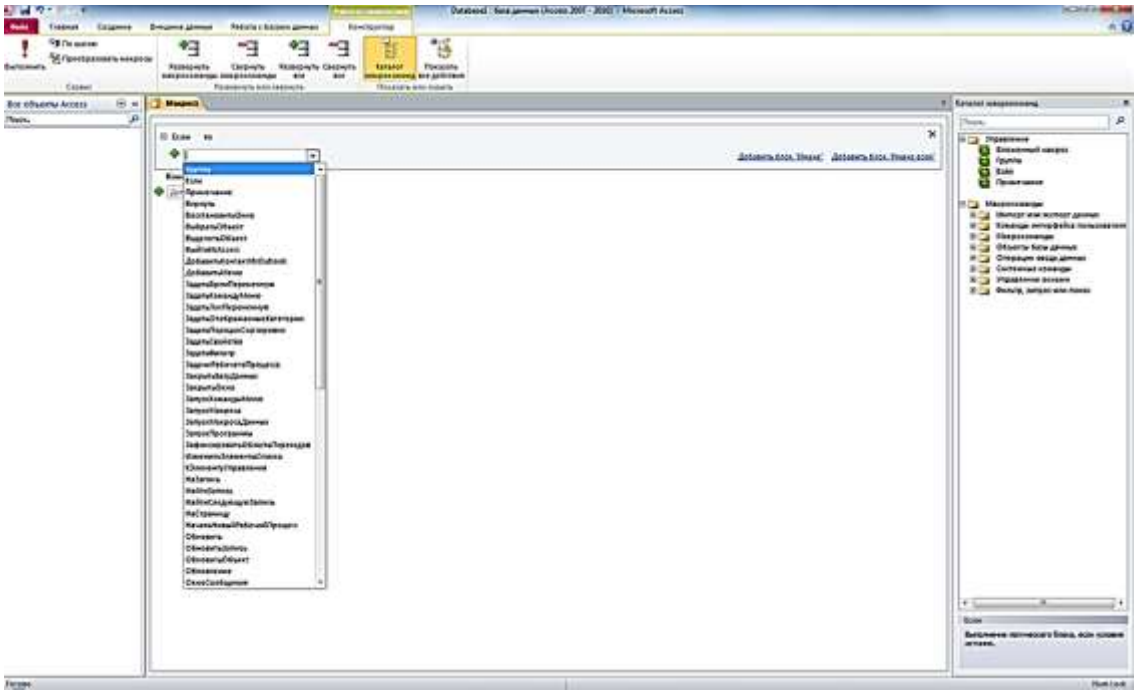


Рис. 6.1. Випадаючий список конструктора макросів

У Access 2010 включено 86 вбудованих макросів. Багато хто з них містить додаткові аргументи для надання їм більшої гнучкості та функціональності. Наприклад, макрос ВідкритиФорму пропонує список всіх зовнішніх форм з бази даних, а також цілий ряд додаткових аргументів. Наприклад, змінюючи аргумент Режим, можна відкривати вибрану форму у режимі форми, а й у режимах конструктор, макет та інших.

У конструкторі макросів попередніх версій Microsoft Access у стовпці Умову можна створювати прості умовні оператори. У конструкторі макросів Access 2010 можна створювати більш універсальні оператори Якщо (If) шляхом додавання операторів Інакше Якщо (ElseIf) та Інакше (Else). Щоб додати ці оператори, виберіть блок ЯКЩО і клацніть у нижньому правому куті блоку коду текст Інакше Якщоабо інакше. Наприклад, якщо клацнути текст Інакше Якщо, відкриється діалогове вікно Інакше Якщо. У міру введення коду в поле умови в програмі Microsoft Access за допомогою технології IntelliSense відображатимуться ідентифікатори, функції та інші елементи бази даних.

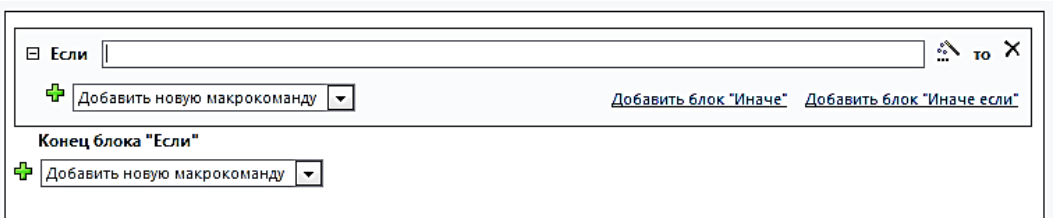


Рис. 6.2. Простий блок Якщо …

Умовний вираз можна записати безпосередньо в поле поруч із командою Якщо або скористатися виробником, натиснувши клавішу праворуч від поля запису умовного висловлювання.

Для додавання нової умови – натисніть гіперпосилання Додати блок «Інакше» або Додати блок «Інакше якщо».

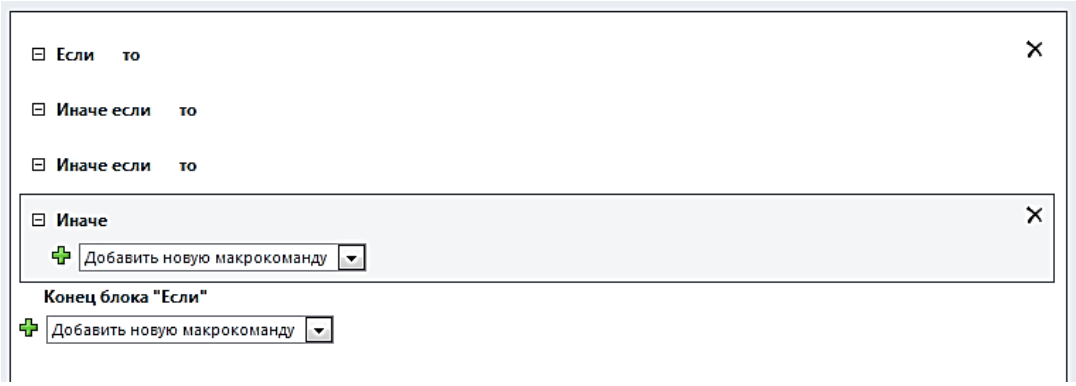


Рис. 6.3. Багаторівневий блок Якщо… Інакше якщо…

6.2 ВКЛАДЕНІ МАКРОСИ.

Вкладений макрос-іменований макрос, включений у простий макрос.

Оператор Вкладений макрос дозволяє визначити окремі набори макрокоманд. Макрокоманди, визначені у вкладеному макросі, можна запустити з іншого макросу за допомогою макрокоманди ЗапускМакросу. Щоб запустити вкладений макрос, у аргументі "Ім'я макросу" макрокоманди ЗапускМакросу використовуйте наступний синтаксис:

<ім'я макросу>.<ім'я вкладеного макросу>

У Access 2010 для включення в макрос вкладеного макросу потрібно вибрати відповідну макрокоманду з списку, що розкривається, в полі Додати нову макрокоманду або перетягнути Вкладений макрос у потрібне місце з каталогу команд. У макросі відобразиться блок, у якому за умовчанням вкладеному макросу присвоєно ім'я Sub1, надається можливість додавання макрокоманд і є ознакою кінця вкладеного макросу.

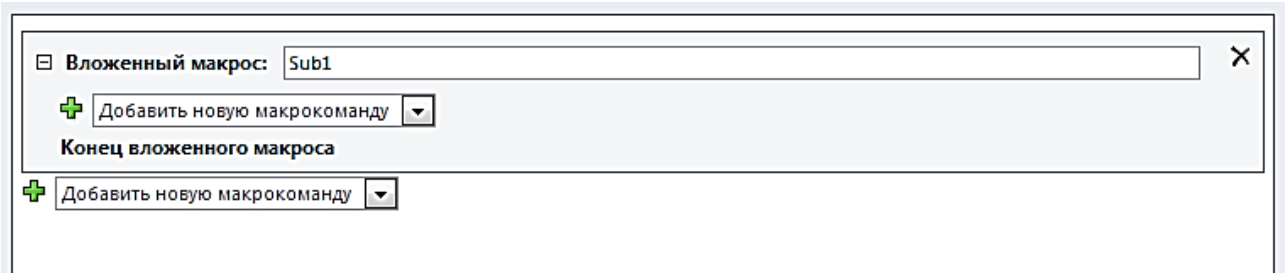
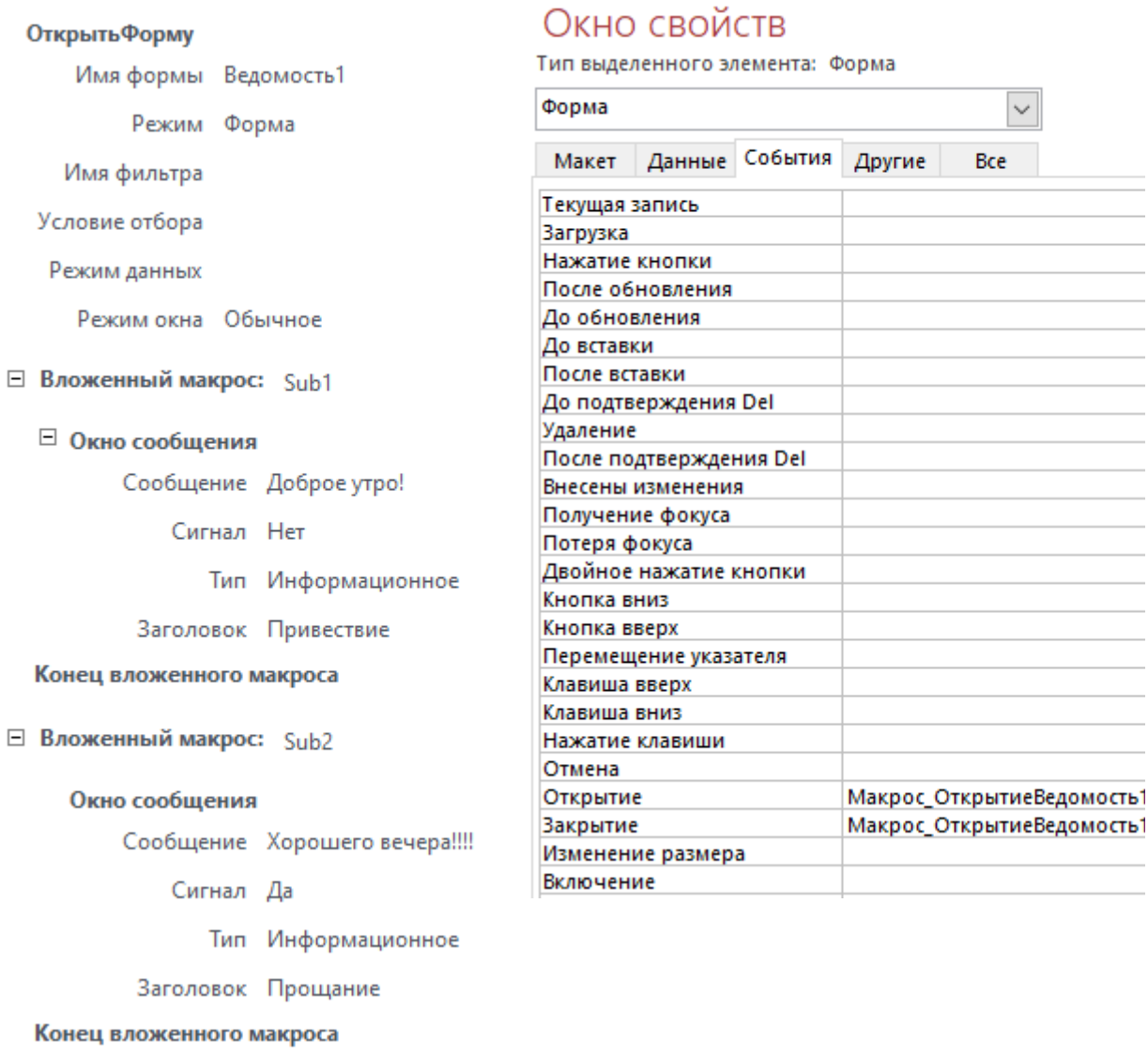


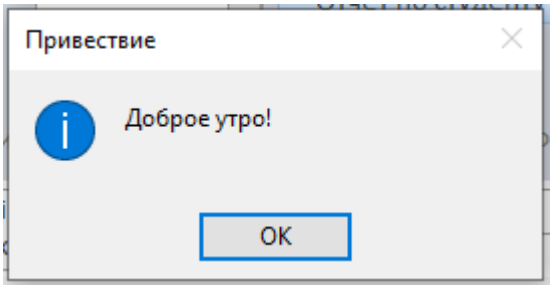
Рис. 6.4. Блок порожнього вкладеного макросу

Приклад :

У ізольований макрос для відкриття форми додати вкладені макроси, які виводять вікно повідомлення при відкритті та закритті форми.



Результат роботи вкладеного макросу на подію Відкриття форми:



6.3 ВПРОВАДЖЕНІ МАКРОСИ.

Впроваджений макрос не відображається в області навігації, однак його можна викликати з таких подій, як Завантаження або Натискання кнопки. Оскільки макрос стає частиною об'єкта форми або звіту, впроваджені макроси рекомендується створювати для автоматизації завдань, які є специфічними для певної форми або звіту.

На впроваджені макроси не можна послатися з інших макросів. Якщо Ви повинні знову використати дії, які визначаються макросом, слід створити новий окремий макрос.

Щоб створити впроваджений макрос, виберіть конструктор макросів у якості події об'єкта управління, потім діалогове вікно конструктора, в якому відзначаєте - Макрос. Як тільки Ви створите макрос, властивість події змінюється на [Впроваджений макрос] (див. рис. 6.5).

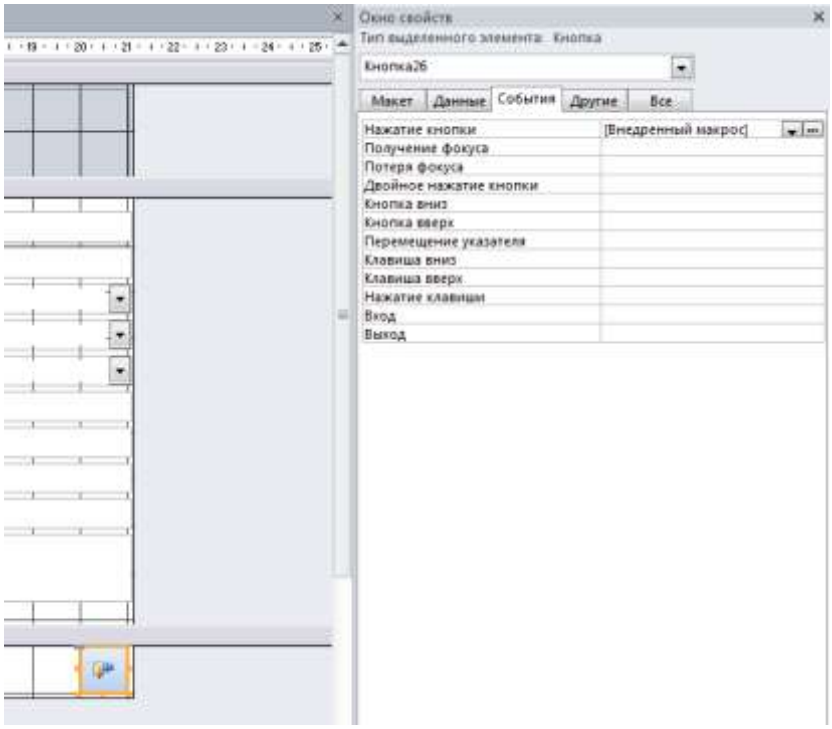


Рис. 6.5. Впроваджений макрос

Впроваджений макрос завжди пов'язується з подією та зберігається у формі чи звіті.

Приклад :

У форму додати елемент керування Поле зі списком, який дозволяє відбирати записи за значенням вибраного списку поля.

На основі запиту на вибірку з кількох таблиць (рис.6.6) створити стрічкову форму.

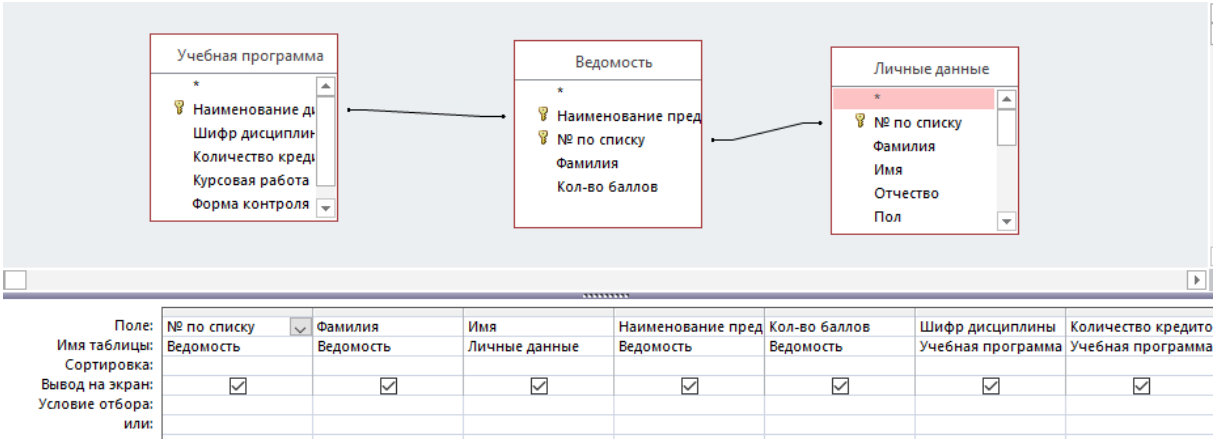


Рис. 6.6.

Потрібно сформувати вибір записів за вибраним полем Найменування предмета використовуючи елемент управління Поле зі списком (рис.6.7):

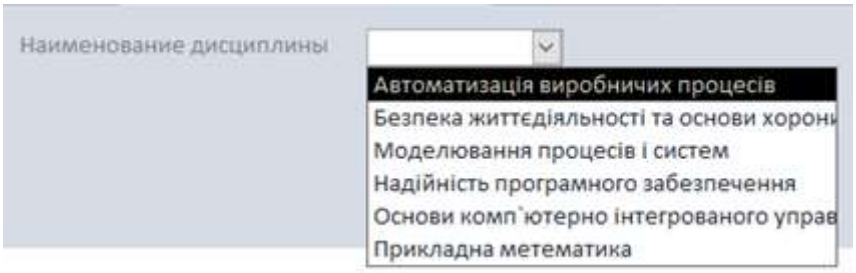


Рис. 6.7.

1. Створимо у конструкторі елемент керування Поле зі списком.

2. У вікні властивостей для поля зі списком (рис.6.8) задаємо джерело рядків. як джерело рядків поле Найменування предмета

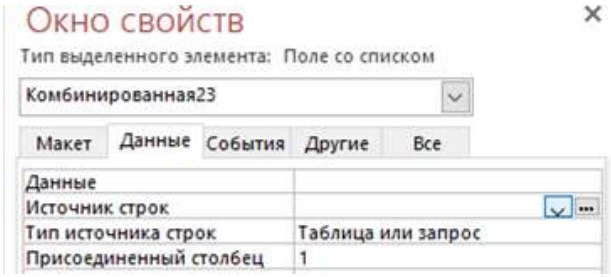
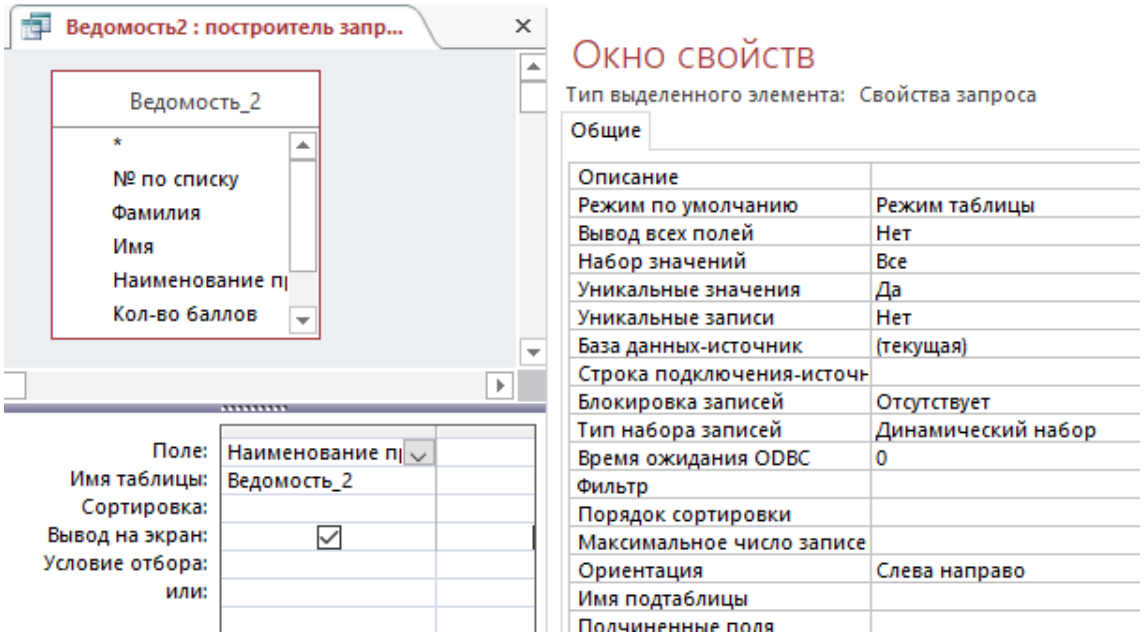


Рис. 6.8.

3. При натисканні на кнопку будівельника відкриється вікно конструктора запиту, на основі якого створено форму. Виберемо в бланк запиту потрібне поле. У разі це - найменування предмета.

У вікні властивостей на вкладці загальні пункту Унікальні значення встановимо значення Так.



Рис, 6.9.

4. Повернемося до конструктора форми. Відкриємо вікно властивостей. Для елемента керування Поле зі списком на вкладці Події з пункту Після оновлення викличемо будівельник макросів (рис. 6.10).

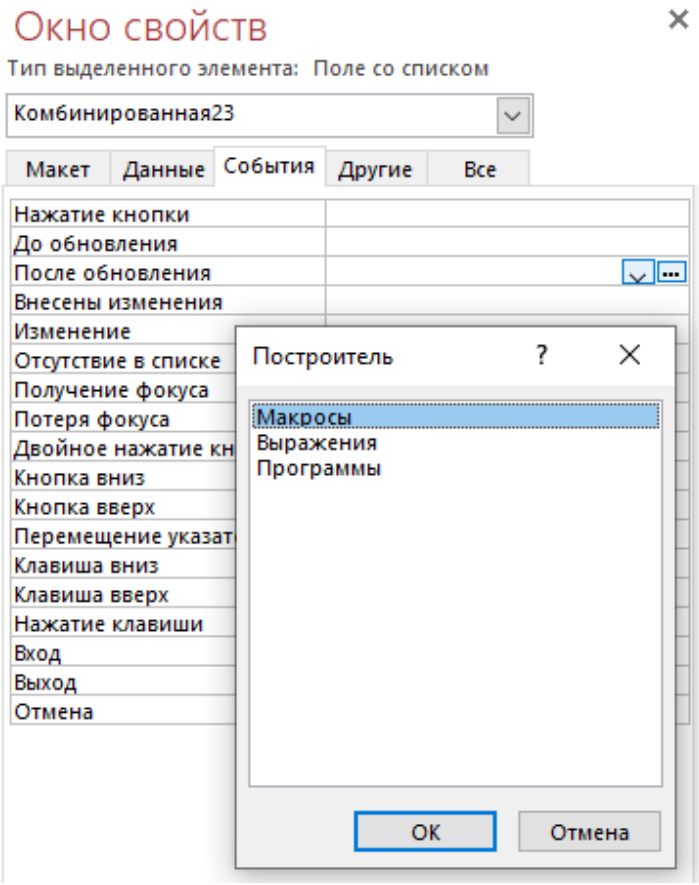


Рис. 6.10.

5. У конструкторі макросів, що відкрився, виберемо макрокоманду Задати фільтр:

ЗадатьФильтр

Имя фильтра

Условие отбора = [Наименование предмета]=[Формы]![Ведомость2]![Комбинированная23]

Имя элемента управления

6. В якості події Після оновлення з'явився [Впроваджений макрос] (Рис.6.11).

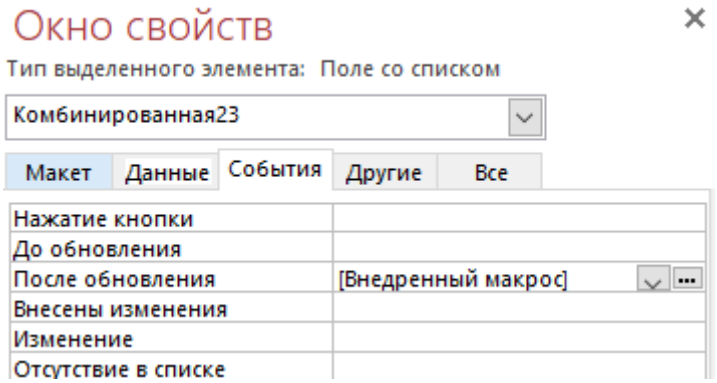


Рис. 6.11.

7. Результат виконання макросу (Рис.6.12):

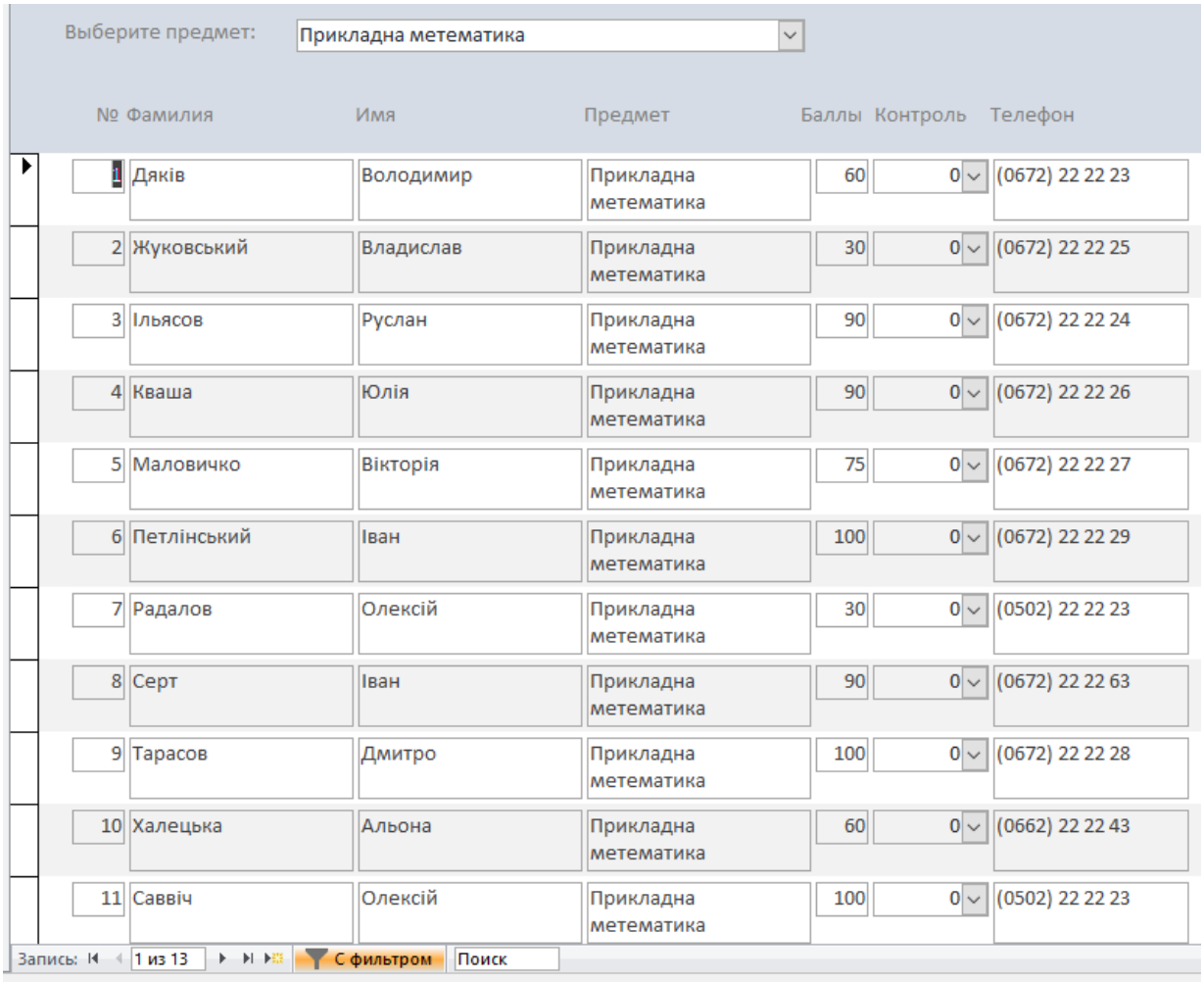


Рис. 6.12.

6.4 МАКРОСИ ДАНИХ. ІМЕНОВАНІ МАКРОСИ.

Макроси даних - це нова можливість у Access 2010. Макроси даних дозволяють прив'язувати логіку до записів та таблиць. При цьому логіка пишеться один раз, а всі форми та код для додавання, оновлення та видалення даних у таблиці успадковують цю логіку. Макроси даних дозволяють реалізовувати різні сценарії.

Макроси даних бувають двох типів: макроси «подій», що спрацьовують при виконанні над даними в таблиці певної дії, і автономні «іменовані» макроси, що запускаються при їхньому виклику на ім'я. Макрос даних можна запрограмувати на запуск відразу після подій додавання, оновлення або видалення даних або безпосередньо перед подією видалення або зміни даних.

Логіка макросу даних працює лише з локальними таблицями, а чи не з пов'язаними таблицями; це обмеження можна обійти з використанням інтерфейсної бази даних Access і бази даних Access з таблицями шляхом додавання макросів даних у вихідну таблицю.

Макроси даних дозволяють уникнути захаращення бази даних завдяки відсутності необхідності пов'язувати той самий макрос з декількома формами. При додаванні логіки таблицю кожна форма, створена основі таблиці, успадковує цю логіку. За допомогою макросів даних можна забезпечити цілісність даних. Припустимо, подія спрацьовує у вигляді, прив'язаної до таблиці без макросу даних. Якщо користувач має доступ до таблиць або може виконувати запити, він може обійти форму, порушивши таким чином логіку. Можна обмежити доступ до таблиць і заборонити виконання запитів, але це не завжди. При додаванні логіки безпосередньо в таблицю макрокоманда спрацьовує, навіть якщо користувач вносить зміни без використання форми.

Події, що підтримуються макросами даних:

Події	**Використовується**
До зміни	Можливі дії: · Видача повідомлення про помилку та блокування змін · Порівняння старого значення з новим та відкриття іншої форми
До видалення	Можливі дії: · Видача повідомлення про помилку та скасування видалення · Висновок форми для зміни порядку проходження елементів
Після оновлення, Після вставки, Після видалення	Можливі дії: · Створення повідомлення електронної пошти · Обхід у циклі набору записів та оновлення їх стану

Щоб створити макрос даних, необхідно відкрити таблицю в режимі таблиці на вкладці Стрічки Робота з таблицями | Таблиця відображаються кнопки створення макросів даних, які будуть виконуватись перед зміною, після оновлення і т.д.. Після натискання відповідної кнопки відкриється конструктор макросів. Для кожної таблиці може бути створено по одному макросу даних на кожну з представлених на Стрічці дій.

Список макрокоманд для макросу даних відрізняється від пропонованого для макросів бази даних. У розділі каталогу макрокоманд У цій базі даних відображаються макроси даних, створені для таблиць.

Макроси даних впроваджуються у таблицю. Якщо для певної дії у таблиці створено макрос даних, то відповідна кнопка на вкладці Стрічки виділяється (рис.6.13).

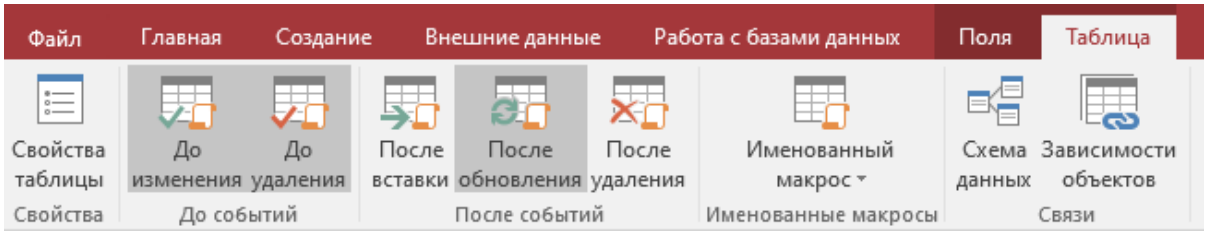


Рис. 6.13.

Приклад :

У вихідну таблицю Відомість, яка мала структуру:

Имя поля	Тип данных
Наименование предмета	Короткий текст
№ по списку	Числовой
Фамилия	Короткий текст
Кол-во баллов	Числовой

Додамо поле Оцінка (текстове поле), яке при заповненні поля Кількість балів, буде приймати відповідно значення: незадовільно(менше 60 балів), задовільно(від 60 до 74 балів)б добре (від 75 до 89 балів), відмінно(90 балів та вище).

Додамо поле Перездача (логічне поле), яке прийматиме значення Істина, якщо кількість балів за предмет менше 60. В інших випадках - Брехня.

Имя поля	Тип данных
Наименование предмета	Короткий текст
№ по списку	Числовой
Фамилия	Короткий текст
Кол-во баллов	Числовой
Оценка	Короткий текст
Пересдача	Логический

Впровадимо макрос даних, який підтримує події До зміни до таблиці Відомість:

```
Если [Кол-во баллов]<60 то
    Группа:
        ЗадатьПоле
            Имя Оценка
            Значение = "неудовлетворительно"
        ЗадатьПоле
            Имя Пересдача
            Значение = Да
    Конец группы
Конец блока "Если"
```

Если [Кол-во баллов]>=60 And [Кол-во баллов]<75 **то**

Группа:

ЗадатьПоле

Имя Оценка

Значение = "удовлетворительно"

ЗадатьПоле

Имя Пересдача

Значение = Нет

Конец группы

Конец блока "Если"

Если [Кол-во баллов]>=75 And [Кол-во баллов]<90 **то**

Группа:

ЗадатьПоле

Имя Оценка

Значение = "хорошо"

ЗадатьПоле

Имя Пересдача

Значение = Нет

Конец группы

Конец блока "Если"

Если [Кол-во баллов]>=90 **то**

Группа:

ЗадатьПоле

Имя Оценка

Значение = "отлично"

ЗадатьПоле

Имя Пересдача

Значение = Нет

Конец группы

Конец блока "Если"

Если [Кол-во баллов] Is Null **то**

ВыводОшибки

Номер ошибки 1

Описание ошибки Заполните баллы

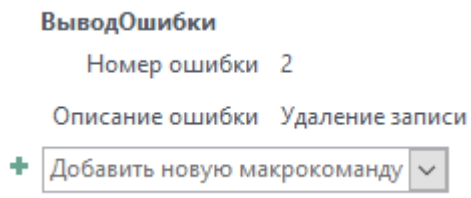
Конец блока "Если"

При заповненні поля Кількість балів, поля Оцінка та Перездавання будуть заповнюватися автоматично:

Наименование предмета	№ по списк	Фамилия	Кол-во баллов	Оценка	Пересдача
Автоматизація виробничих процесів	1	Дяків	75	хорошо	☐
Автоматизація виробничих процесів	2	Жуковський	90	отлично	☐
Автоматизація виробничих процесів	3	Ільясов	75	хорошо	☐
Автоматизація виробничих процесів	4	Кваша	59	неудовлетвор	☑
Автоматизація виробничих процесів	5	Маловичко	90	отлично	☐
Автоматизація виробничих процесів	6	Петлінський	75	хорошо	☐
Автоматизація виробничих процесів	7	Радалов	75	хорошо	☐
Автоматизація виробничих процесів	8	Серт	90	отлично	☐
Автоматизація виробничих процесів	9	Тарасов	30	неудовлетвор	☑
Автоматизація виробничих процесів	10	Халецька	40	неудовлетвор	☑

Приклад :

Впровадимо в таблицю Відомість макрос даних, який підтримує події До видалення, який видає повідомлення про помилку та скасовує видалення:



Названі макроси даних.

Крім макросів даних, пов'язаних з відображеними на вкладках діями, Access 2010 можуть бути створені іменовані макроси даних. На відміну від макросів даних у цьому, що у останніх можуть визначатися параметри. Значення параметрів надається у зухвалому макросі.

Щоб створити іменований макрос даних, в області навігації зліва двічі клацніть ім'я таблиці, до якої необхідно прив'язати макрос. На вкладці Таблиця групи Іменовані макроси натисніть кнопку Іменований макрос і виберіть команду Створити іменований макрос (рис 10.2).

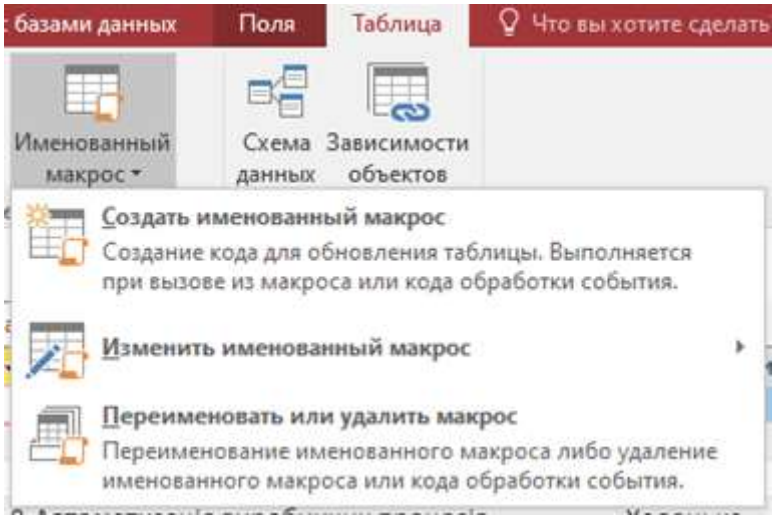


Рис. 6.14. Список команд іменованого макросу

Перша команда списку, що випадає, дозволяє створити іменований макрос, друга - змінити іменований макрос, третя - переглянути всі таблиці, що містять макроси.

Під час створення іменованого макросу за умовчанням йому присвоюється ім'я МакросДанных1, при зверненні щодо нього доповнюється ім'ям таблиці, де він створено, тобто. повне ім'я: Ім'я Таблиці. МакросДаних1.

Запуск іменованого макросу здійснюється макрокомандою ЗапускМакросаДанних, доступною як з макросу даних, так і з будь-яких інших макросів програми. Наприклад, працюючи у формі, можна виконувати обчислення через звернення до полів форми, а викликати іменований макрос, який виконає обчислення у таблиці і зміни відразу відобразяться у формі.

Командою Перейменування та видалення макросів відкривається вікно диспетчера макросів даних (рис. 6.15). У ньому перераховані всі таблиці основи, прив'язані до них макроси, названі макроси.

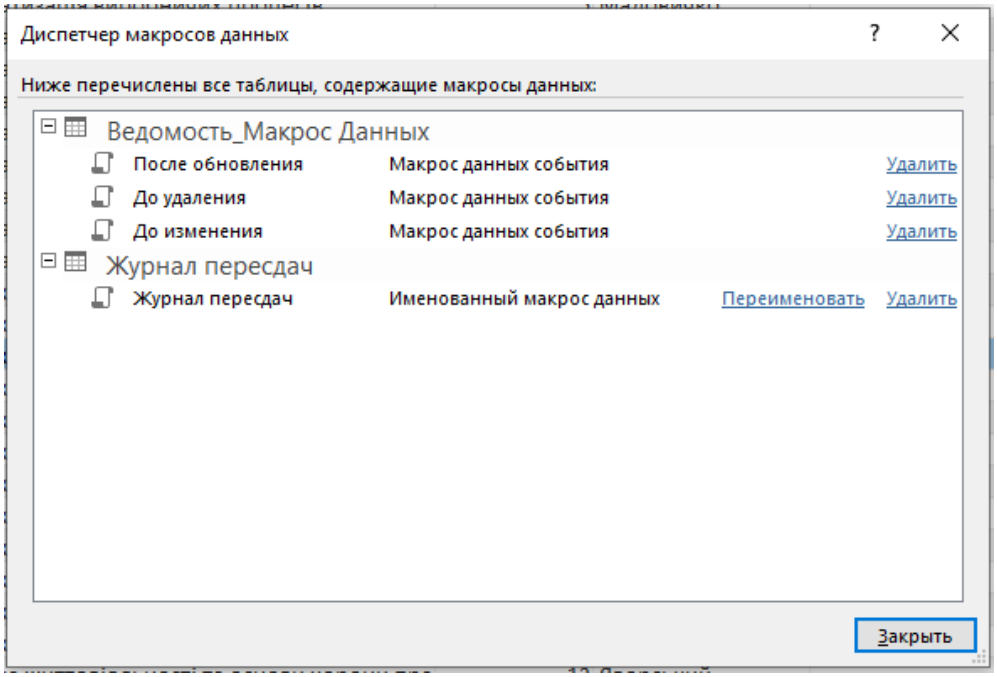


Рис. 6.15. Список таблиць, що містить макроси даних

Ще одна можливість, підтримувана названими макросами даних - це можливість передачі параметрів. Параметри часто використовуються для підвищення продуктивності бази даних, оскільки дозволяють розробникам і користувачам примусово обмежити набори даних перед виконанням макросу. Це дозволяє підвищити швидкість роботи макросів, знизити навантаження на сервери баз даних та зменшити мережевий трафік. Параметри також забезпечують додаткову гнучкість, оскільки дозволяють повторно використовувати той самий макрос без змін. Щоб додати параметри до іменованого макросу даних, двічі клацніть таблицю, до якої необхідно прив'язати макрос. На вкладці Таблиця в групі Іменовані макроси клацніть розкривний список Іменований макрос і виберіть команду Створити іменований макрос. У верхній частині конструктора макросів (рис. 6.16).

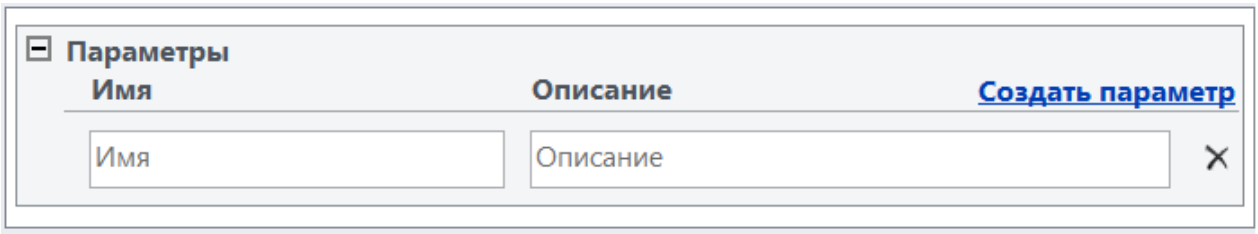


Рис. 6.16. Команда "Створити параметр" у конструкторі макросів

За допомогою параметрів іменований макрос передаються значення, які потім можна використовувати в умовних операторах або в розрахунках. Крім того, за допомогою параметрів із звичайних макросів макрос даних передаються посилання на об'єкти.

У полі Ім'я введіть унікальне ім'я параметра. Це ім'я буде використано для вказівки параметра у виразах.

Якщо потрібно, введіть опис параметра у вікні Опис. Рекомендується зробити це, оскільки пізніше при використанні параметра текст опису буде відображатися як підказки. Це допоможе запам'ятати призначення параметра.

Приклад :

Створимо таблицю Журнал пересдач, в яку за допомогою впроваджених макросів заноситиметься інформація про перескладання.

Створимо порожню таблицю Журнал перездач, яка має таку структуру:

Имя поля	Тип данных
№ записи	Счетчик
Наименование предмета	Короткий текст
Фамилия	Короткий текст
Кол-во баллов старое	Числовой
Кол-во баллов новое	Числовой
Дата пересдачи	Дата и время

Полю № запису задамо тип Лічильник, що дозволить автоматично привласнювати номери записів, що створюються.

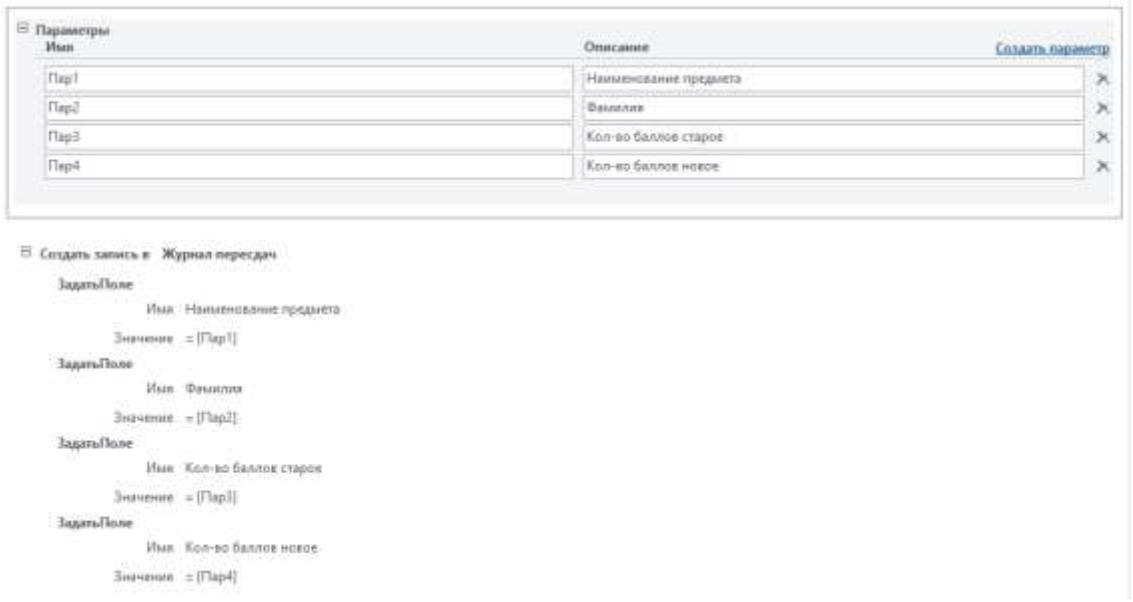
У властивостях поля Дата перездачі встановимо значення за замовчуванням =Дата(), що дозволить автоматично заповнювати поле Дата перездачі поточною датою:

Свойства поля

Общие | Подстановка

Формат поля	
Маска ввода	
Подпись	
Значение по умолчани	=Дата()
Правило проверки	
Сообщение об ошибке	
Обязательное поле	Нет
Индексированное поле	Нет
Режим IME	Нет контроля
Режим предложений IM	Нет
Выравнивание текста	Общее
Отображать элемент вы	Для дат

Створимо іменований макрос і назвемо його.



Повернемося до таблиці Відомість тавпровадимо макрос даних, який підтримує події Після оновлення:

```
Если Updated("Кол-во баллов")=Истина то
    ЗапускМакросаДанных
     Имя макроса  Журнал пересдач.Журнал пересдач
    Параметры
          Пар1  = [Наименование предмета]
          Пар2  = [Фамилия]
          Пар3  = [Old].[Кол-во баллов]
          Пар4  = [Кол-во баллов]
Конец блока "Если"
```

У блоці Якщо в логічному вислові умова чи змінювалося значення в полі Кількість балів перевіряється за допомогою функції Updated (“ім'я поля”).

При істинному значенні висловлювання макрокомандою ЗапускМакросаДанных запускається виконання створений раніше іменований макрос даних Журнал персдач.Журнал пересдач.

У блоці Параметри натиснути спочатку Оновити параметри в нижньому правому куті, потім присвоїти кожному параметру значення з оновленого запису таблиці Відомість. Закрити макрос, зберігаючи всі зміни.

В результаті при зміні значення поля Кількість балів у таблиці Відомість, в Журнал пердач додається новий запис.:

№ запису	Найменування предмета	Фамилия	Кол-во баллов старое	Кол-во баллов новое	Дата перерахунку

…

9	Безпека життєдіяльності та основи хорони пра	Жуковський	65	80	19.11.2020
10	Безпека життєдіяльності та основи хорони пра	Ільясов	70	90	19.11.2020
11	Безпека життєдіяльності та основи хорони пра	Кваша	60	75	19.11.2020
12	Безпека життєдіяльності та основи хорони пра	Маловичко	30	60	19.11.2020
(№)			0	0	19.11.2020

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1. Проектування баз даних у середовищі MS ACCESS 2010. Навчальний посібник / О. М. Верес, І. В. Рішняк. Львів : Видавництво Львівської політехніки, 2016. - 232 с.

2. Бекаревич, Ю. Б. Самоучитель Access 2016 / Ю. Б. Бекаревич, Н. В. Пушкина. — СПб.: БХВ-Петербург, 2016. — 432 с.:

3. Гурвиц Г. MS Access 2013. Разработка приложений на реальних примерах / Г. Гурвиц. - СПб. : Питер, 2014. - 497 с.

4. Аблязов В.И . Проектирование баз данных в среде Microsoft Office Access 2003, 2007 и 2010./Аблязов В.И, Издательство Политехнического университета, - 2014,- 107 с.

5. Литвин В.В. Бази знань інтелектуальних систем підтримки прийняття рішень. Монографія. Львів: Видавництво Львівської політехніки, 2011. - 240 с

6. Жежич П. І. Консолідовані інформаційні ресурси баз даних та знань : навчальний посібник / П. І. Жежич ; ред. В. В. Пасічник. - Львів : Видво Львівська політехніка, 2010. - 212 с.

7. Федько В. В. Створення баз даних та застосувань професійного спрямування. Лабораторний практикум: Навч.-практ. посібн. / Укл. В. В. Федько, В. І. Плоткін. – Харків : Вид. ХНЕУ, 2009.– 208 с.

8. Шпортько А., Шпортько Л. Розробка баз даних в СУБД Microsoft Access: Практикум для студентів вищих та учнів професійно-технічних навчальних закладів/О.В.Шпортько, Л.В. Шпортько. – К.: Видавничий дім «Кондор», 2018. – 184с.