Міністерство освіти і науки України
Сумський державний педагогічний університет ім. А.С. Макаренка

Кафедра інформатики
Шамшина Н.В.

ІНФОРМАТИКА.
СИСТЕМА УПРАВЛІННЯ БАЗАМИ ДАНИХ MICROSOFT ACCESS

Навчальний посібник

Суми-2015

УДК 004.65+371.67/.69
ББК 30.973.2я73
П69

Рекомендовано до друку рішенням редакційно-видавничої ради Сумського державного педагогічного університету ім. А.С. Макаренка

Рецензенти: **Шамоня В.Г.,** кандидат фізико-математичних наук, доцент Сумського державного педагогічного університету ім. А.С. Макаренка;
Семеніхіна О.В., кандидат педагогічних наук, доцент, зав. каф. інформатики Сумського державного педагогічного університету ім. А.С. Макаренка

Шамшина Н.В.
П69 Інформатика. Система управління базами даних Microsoft Access. Навчальний посібник. – Суми: Вид-во СумДПУ ім. А.С. Макаренка, 2015. – 72 с.

Навчальний посібник призначений для студентів фізико-математичних факультетів педагогічних університетів для організації самостійної та індивідуальної роботи при вивченні системи управління базами даних MS Access.

УДК 004.65+371.67/.69
ББК 32.973.2я73

©Шамшина Н.В., 2015
©СумДПУ ім. А.С. Макаренка, 2015

Зміст

Вступ 4

Розділ 1. ТЕОРЕТИЧНІ ОСНОВИ РОБОТИ СУБД ACCESS 5

1.1. Загальні відомості з теорії баз даних 5
1.2. Структура бази даних СУБД Access 5
1.3. Типи полів в базі даних Access 9
1.4. Властивості полів в базі даних Access 11
1.5. Вирази в Microsoft Access 13
1.6. Функції у Microsoft Access 18
1.7. Особливості зберігання та обробки інформації в базах даних 22

Розділ 2. ПРАКТИЧНІ ОСНОВИ РОБОТИ СУБД ACCESS 23

Початок роботи з Microsoft Access 23
2.1. Створення бази даних Access 25
Лабораторна робота №1. Створення бази даних 29
2.2. Сортування, пошук, фільтрація записів бази даних 31
Лабораторна робота №2. Робота з базою даних у режимі таблиці 34
2.3. Запити 36
Лабораторна робота №3. Створення запитів на вибірку 40
2.4. Створення групових запитів та запитів на зміну 41
Лабораторна робота №4. Створення групових запитів та запитів на зміну 46
2.5. Форми 47
Лабораторна робота №5. Створення форм 50
2.6. Звіти 52
Лабораторна робота №6. Створення звітів 55
2.7. Організація інтерфейсу для роботи з базою даних Access. Макроси . 56
Лабораторна робота №7. Створення кнопкової форми та макросів 59

Додаток А. Приклади запитів SQL й QBE 61

Додаток Б. Основні інструкції мови SQL 70

Предметний покажчик 71

Література 72

Вступ

Система управління базами даних (СУБД) відноситься до класу прикладного програмного забезпечення загального призначення і може бути використана багатьма користувачами незалежно від фахової освіти. За допомогою СУБД можна зберігати великі організовані масиви даних, здійснювати аналіз даних, автоматизувати обробку даних в офісі. Згадані чинники зумовили включення до навчальних планів підготовки спеціальностей фізико-математичного та природничого напрямку модуль, присвячений вивченню СУБД.

Основною метою вивчення цього розділу інформатики є знайомство з типовим набором функцій та можливостями системи управління базами даних щодо використання в майбутній професійній діяльності. СУБД Microsoft Access (MS Access) входить у широко розповсюджений пакет Microsoft Office, встановлюється практично на кожний персональний комп'ютер, тому вивченню саме цього середовища присвячене навчальне видання.

Посібник містить стислий теоретичний матеріал з теорії реляційних баз даних та правил використання програми MS Access, який супроводжуються практичними завданнями та вправами, що сприяють засвоєнню набутих знань, умінь і навичок. У першому розділі розглянуті теоретичні питання створення та проектування баз даних, правил роботи з даними у таблицях, використання функцій для роботи з різними типами даних. Другий розділ присвячений описанню виконання практичних завдань зі створення структури баз даних, пошуку та аналізу даних у таблицях, створення запитів, форм, звітів, а також організації інтерфейсу для роботи з базами даних. Детально розглянуто наочні приклади, проілюстровані задачами відповідно до програми з інформатики для вищих навчальних закладів. До кожної теми розроблені лабораторні роботи з оригінальними завданнями, які дозволяють поетапно засвоїти матеріал курсу. На допомогу студенту посібник містить у додатку розв'язки завдань з лабораторних робіт № 3,4 – ілюстрації запитів різного типу, а також предметний покажчик.

Умовні позначення:

✍ зверніть увагу, це важливо

✎ приклади

☞ *Вказівки.* пояснення до виконання самостійних завдань

? питання для контролю засвоєння теми.

Розділ 1. ТЕОРЕТИЧНІ ОСНОВИ РОБОТИ СУБД ACCESS

1.1. Загальні відомості з теорії баз даних

База даних **(БД) –** сукупність взаємозв'язаних даних про групу об'єктів, містить інформацію необхідну для автоматизації роботи фірми або установи. Наприклад: БД книжкового фонду бібліотеки, БД кадрового складу підприємства та ін.

Система Управління Базами Даних (**СУБД**) – програма, призначена для організації, зберігання і пошуку інформації в БД, забезпечує використання БД різними користувачами.

БД зберігаються у файлах даних. Файл даних **(ФД) –** послідовність записів певної структури, ФД зберігаються на електронних носіях – дисках, і мають спеціальний формат. За способом організації зв’язків між елементами запису БД, за структурою записів БД класифікують по типах.

Типи (Моделі) БД:

1. ієрархічна
2. мережева
3. реляційна
4. багатовимірна

Ієрархічна БД – відносини між елементами записів мають дерево-подібну (ієрархічну) структуру. Елементи такого дерева відносин називають вузлами. Кожен вузол підкоряється тільки одному вище стоячому, на найвищому рівні – вузол, що не підкоряється жодному, тобто корінь дерева. Типовий приклад – файлова структура на диску з каталогами різного рівня. Пошук даних при цьому вельми трудомісткий – необхідно послідовно, крок за кроком, пройти всі підлеглі каталоги, переміщаючись по гілках дерева каталогів.

Мережева БД – має гнучкішу організацію, кожен елемент запису може мати довільні зв'язки з будь-яким іншим елементом, до вертикальних зв'язків додаються горизонтальні. Пошук елементів полегшується.

Реляційна БД – найпоширеніший тип. Інформація організована у вигляді двовимірної таблиці, однієї або декількох взаємозв'язаних, які створюють структуру БД. Зв'язок між елементами – записами здійснюється по загальних полях.

Багатовимірні БД – на відміну від реляційних 2-мірних, використовують для зберігання даних 3-х, 4-х або N-мірний куб. Застосовуються для аналізу та прогнозування.

Сучасні БД використовують об'єктно-орієнтовану технологію, коли дані і код для їх обробки упаковані разом. При цьому знімаються обмеження на типи даних і їх довжину.

1.2. Структура бази даних СУБД Access

Програма СУБД MS Access є сучасною реляційною БД та входить в пакет MS OFFISE. СУБД MS Access призначена як для створення і роботи

невеликих БД (настільних), так і могутніх мережевих застосувань архітектури клієнт-сервер.

Файли БД MS Access мають формат *.mdb (версія 2003, 2007), або *.accdb (версія 2010, 2013) У файлах містяться об'єкти (таблиці, запити, форми і т.д.) і програмний код для їх обробки, необхідний для зміни та перегляду даних.

БД Access має, як правило, декілька об'єктів, які взаємодіють між собою. Об'єкти різного типу і виконують різні функції (табл. 1).

Таблиця 1.

Типи об'єктів Access

№	Об'єкт	Функція	Характеристика
1.	Базові таблиці зв'язані між собою	Організація даних	Основний об'єкт для зберігання даних, заповнюються у міру надходження інформації, зберігаються у файлі даних на диску. У режимі перегляду дані можна вводити, редагувати, здійснювати пошук, сортування, фільтрацію.
2.	Форми	Введення і редагування даних	Зручний і наочний засіб для роботи з даними, альтернатива табличному режиму роботи. Дозволяє вводити і редагувати дані одночасно з декількох зв’язаних таблиць, містить засоби автоматизації, елементи керування. У файлі даних на диску зберігається макет форми, при відкриванні у полях форми з’являються дані з таблиць, які вказано як джерело даних у макеті. У режимі перегляду дані можна вводити, редагувати, здійснювати пошук, сортування, фільтрацію.
3.	Запити	Відбір і аналіз даних, розрахунки за формулами	Основний інструмент автоматизації обробки даних. Типи запитів різноманітні та суттєво відрізняються. Найбільш поширеними є запити на вибірку – це віртуальні таблиці, які містять дані, витягнуті з базових таблиць по запитах користувача. У файлі даних на диску зберігається інструкція запиту, при відкриванні у полях запиту з’являються дані з таблиць, які вказано як джерело даних в інструкції. Складні запити містять віртуальні поля, де йде обробка даних, обчислення за формулами. Можливі вкладені запити.
4.	Звіти	Представлення даних	Документ готовий до друку на принтері. Містить підсумкові дані, графіки і діаграми, елементи оформлення і форматування друкарського документа. У файлі даних на диску зберігається макет звіту, при відкриванні у звіті з’являються

			дані з таблиць, які вказано як джерело даних в макеті.
5	Макроси	Автоматизація роботи з даними (прості операції)	Послідовність макрокоманд, призначені для створення елементів керування у формах. Спрощують створення розвинутого інтерфейсу для роботи з базою даних.
6.	Модулі	Автоматизація роботи з даними	Підпрограми, написані на мові VBA (*Visual Basic for Applications*), функції, процедури які виконують складну обробку даних.
7.	Сторінки	Зв'язок з Internet	Сторінки доступу до даних, об'єкт HTML, містять компоненти через які здійснюється зв'язок WEB-сторінки з БД на сервері.

Створення БД включає два етапи: створення структури БД та заповнення БД записами. Створити структуру БД означає створити базові таблиці, та зв'язати їх між собою. Після тестування БД можна заповнювати даними. Заповнення БД записами здійснюється з надходженням інформації.

Базові таблиці – основа БД. Базові таблиці описують групу об’єктів та складаються з наборів записів однакової структури. У теорії реляційних БД рядки таблиці – це записи, стовпці таблиці – поля. Можна сказати, що таблиця – це двовимірний масив, де рядки утворені окремими записами (records), а стовпці – полями (fields) цього запису.

Запис **(**RECORD) – одна з багатьох, однотипних по своїй структурі частин файлу БД, яка характеризує об'єкт з усіх боків важливих для даного завдання.

Записи в свою чергу складаються з ***полів*** **(**FIELD) – кожне з яких характеризує одну особливість об'єкта. Поля можуть мати різний тип відповідно до особливостей об'єкта (числовий, символьний, логічний, типу календарної дати тощо).

Будь-який об'єкт в таблиці унікальний, повторень бути не повинно. Серед всіх полів, що описують об'єкт має бути унікальне ***поле – ключ,*** що однозначно характеризує об'єкт. По ключовому полю здійснюється ідентифікація об'єкта. Найчастіше це поле Код-запису типу «*счетчик*», може бути також інвентарний номер, номер студентського квитка тощо. Ключ може бути простим або складним, який складається з декількох ключових полів. Поле або набір полів, що гарантують унікальність запису має назву ***первинний ключ***.

Як правило в БД не одна таблиця, а декілька зв'язаних між собою. Тому існує поле-***зовнішній ключ*** – це поле, по якому організовується зв'язок між таблицями. У останньому випадку поле–первинний ключ однієї таблиці є також полем в іншій таблиці.

Записи в таблиці містять однакове число полів, тобто таблиця строго прямокутна. Якщо при цьому більшість полів не заповнена, багато даних повторюється, організовують підлеглі таблиці різного рівня. Крім того,

прийнято, щоб окрема таблиця заповнювалася на одному робочому місці. Щоб оптимізувати таблиці у БД проводять нормалізацію таблиць.

Нормалізація – це перенесення з таблиць даних, що повторюються, в нові таблиці, записи яких не містять значень, які повторюються. Після цього за допомогою однакових значень полів між новими таблицями встановлюють зв'язки – так утворюється структура реляційної БД.

У теорії СУБД відомі чотири варіанти зв'язків, традиційно званих відносинами, вони встановлюють правила відповідності записів в двох таблицях, причому «розглядаємо» як би з боку першої таблиці, вона є «головною», а друга «підлеглою».

Відношення **один–до–одного**: одному запису першої таблиці відповідає не більше одного запису другої таблиці і навпаки. Поле – первинний ключ першої таблиці пов'язано з таким самим полем в другій таблиці, яке також є первинним ключем. В обох полях не повинно бути значень, що повторюються, значення повинні відповідати один одному. Можна сказати, що це не дві таблиці, а одна, розділена надвоє вертикальною межею, так що частина полів опинилася в першій таблиці, а частина в другій. Розділення йде за функціональною ознакою, з метою введення даних на окремих робочих місцях.

Відношення **один–до–багатьох**: в цьому випадку одному запису першої таблиці може відповідати декілька записів другої таблиці. Поле – первинний ключ першої таблиці пов'язано з таким самим полем в другій таблиці, яке є зовнішнім ключем, значення по зовнішньому ключу можуть повторюватися. Цей найбільш поширений тип зв’язку використовують задля усунення багатьох повторень у таблиці.

Відношення **багато–до–одного**: декільком записам першої таблиці може відповідати не більше одного запису з другої таблиці. Можна сказати, що це відношення один–до–багатьох навпаки, головною тут є таблиця з боку багатьох записів.

Відношення **багато–до–багатьох**, декільком записам першої таблиці відповідає декілька записів другої таблиці. В цьому випадку зв'язки між таблицями непрямі, організовані за допомогою ще однієї таблиці, з використанням складного ключа, тобто набору ключових полів.

✍ Створити таблицю, або структуру таблиці означає визначити імена, типи і довжину полів в таблиці, для полів числового типу необхідно також вказати точність. Поля указують в такому порядку, щоб на початку таблиці відображалися найбільш важливі особливості об'єктів, ключове поле, як правило, на першому місці.

Для забезпечення швидкості обробки інформації, процедури пошуку, сортування в БД використовують ***індексовані*** поля. У разі БД індексація – це впорядкування таблиці по якому не-будь полю, і створення спеціальної "таблиці–індекса", де записані значення поля відповідно до індексів. Таким чином індекс – це «внутрішня» таблиця, яка упорядковує записи. Для первинного ключа індекс створюється автоматично. Для індексованих полів

використовують алгоритм двійкового пошуку записів, що прискорює процедуру пошуку, особливо для великих БД.

1.3. Типи полів в базі даних Access

Головний принцип побудови реляційної БД – в кожному полі таблиці повинні міститися дані тільки одного типу. Різноманітність властивостей описуваних об'єктів привела до появи багатьох типів і підтипів полів БД, їх ділять на 5 основних груп:

I. Символьні

До цієї групи входять два поля, призначені для введення текстової інформації:

1) тип поля текстовий *Короткий текст* – містить будь-які символи і знаки, довжина від 1-255 символів, замовляється максимально необхідна довжина поля (про економію можна не думати, оскільки при зберіганні інформації на диску всі хвостові пробіли відрізуються). Найпоширеніший тип даних, використовується для даних, які не вимагають математичної обробки: прізвище, ім'я, адреса, номер телефону, номер студентського квитка тощо. Встановлюється за замовчанням, значення за замовчанням «пусто», може бути ключовим та індексованим.

2) тип поля Memo-примітка *Длинный текст* – містить примітки для окремих записів, може бути завдовжки до 65 535 символів, не може бути ключовим та індексованим.

II. Числові

Найчисленніша група, призначена для зберігання чисел, дат та інших даних які підлягають математичній обробці, містить 4 типи полів:

1) *тип поля числової* – для зберігання чисел, найпоширеніший тип полю. Залежно від розміру поля виділяють 5 підтипів числових полів (табл. 2). Розмір поля визначає спосіб зберігання чисел в пам'яті і на диску, також визначає діапазон даних, що зберігаються у ньому.

Таблиця 2.

Підтипи числових полів

№	Підтип	Розмір поля в байтах	Точність	Діапазон значень
1	Байт	1	–	0 – 255
2	Ціле	2	–	-32 768 – 32 768
3	Довге ціле	4	–	$-2 *10^{9} –2 *10^{9}$
4	Одинарне з плаваючою крапкою	4	7	$-3,4 *10^{38} – 3,4 *10^{38}$
5	Подвійне з плаваючою крапкою	8	15	$-1 *10^{308} – 1 *10^{308}$

Необхідно вибирати такий розмір поля, щоб в ньому помістилися будь-які очікувані дані і при цьому займали мінімальну кількість байт. При обчисленнях дані цілого типа обробляються швидше. За замовчанням привласнюється тип довге ціле, значення 0, може бути ключовим та індексованим.

2) *тип поля грошовий* – спеціальний тип поля для роботи з даними, що відображають вартість, ціну і т.д. Розмір поля 4 байти, точність 4, діапазон - $9{,}22 *10^{14}$ – $9{,}22 *10^{14}$, це числа з фіксованою крапкою, значення яких округляються за спеціальними правилами до 4-х знаків після коми. Значення за замовчанням 0, може бути індексованим.

3) *тип поля лічильник (счетчик)* – числове поле що автоматично змінюється при додаванні записів в таблицю, за замовчанням збільшується на 1, але може бути встановлений режим генерації випадкових чисел і автоматичне привласнення цих значень полю лічильник. Розмір поля довге ціле, 4 байти, точність відсутня, діапазон значень від $1 – 2*10^9$.

✍ Поле лічильник у таблиці може бути лише одне! Поле лічильник використовується як код–номер запису, при створенні таблиці з'являється автоматично та є ключовим. Дані в цьому полі не редагуються! Якщо при додаванні записів деякі з них було відмінено або видалено, їх номери відсутні. Це легко виправити після тестування таблиці шляхом видалення поля та створення такого ж самого знову. Якщо потрібно використати поле лічильник для встановлення зв'язку з іншими таблицями уважно слідкуйте за послідовністю номерів.

4) *тип поля логічний* – використовується для опису наявності або відсутності яких не будь ознак, розмір поля 1 байт. Відсутність ознаки задається значенням 0, інтерпретується як Брехня: Ложь, False, No, Нет, Выкл. Присутність ознаки задається значенням -1, інтерпретується як Істина: Истина, True, Yes, Да, Вкл. Не може бути ключовим, індексувати можна.

III. Дата і час

Тип поля дата/час – для зберігання календарних дат і часу, довжина 8 байт, спеціальний формат, тобто число, в якому ціла частина представляє дату, а дробова час. На екран виводиться в зручному для користувача форматі: *11.09.02,* або *11 вересня 2002,* або *11–сен–2002*. За замовчанням має значення «пусто». З датами можливі арифметичні операції (віднімання, складання), операції порівняння. Може бути індексованим і ключовим.

IV. Поле об'єкта OLE – для зберігання растрової і векторної графіки, аудіофайлів, презентаций тощо. Все, що не є текстовою і числовою інформацією може зберігатися в цьому полі. Довжина до 1 Гігабайта, не може бути індексованим і ключовим. Для створення OLE-об'єктів на комп'ютері повинні бути встановлені відповідні програми.

V. Тип поля Гіперпосилання – для зберігання адрес WEB-сторінок розташованих в Internet або локальних мережах, довжина від 1 до 6 144 символів. При клацанні на цьому полі автоматично завантажується документ

за вказаною адресою, це можуть бути документи на вашому комп'ютері, наприклад Word або Excel. Не може бути індексованим і ключовим.

VI. Тип поля Вкладення (*Вложение*) для Access 2010,2013 – для зберігання декількох файлів різних типів у одному полі: документів, презентацій, зображень тощо. Подібно файлам вкладеним в повідомлення електронної пошти позначається зображенням канцелярської скріпки, в рядок заголовку поля вкладення не можна додати текст при створенні поля у режимі таблиці. Кількість вкладених файлів вказується у дужках після канцелярської скріпки. Загальний об’єм файлів до 2 Гігабайт, це максимальний об’єм БД Access. Розмір окремих файлів до 256 Мегабайт. Файли будуть стиснені програмою Access, якщо вже не стиснути іншою програмою, наприклад, як у форматі JPEG. Відкривати файли потрібно подвійним клацанням миші або через контекстне меню. Управління вкладеними файлами здійснюють через діалогове вікно *Вложения*. Поле не може бути індексованим і ключовим.

VI. Тип поля Обчислюване (*Вычисляемый*) для Access 2010,2013 – для зберігання значень, які обчислюються через інші значення. Вирази для обчислення можуть містити значення з полів тієї ж таблиці, а також вбудовані функції Access, не можна використовувати поля з інших таблиць або запитів. На початку виразу не потрібно вводити знак рівності (=). Результати обчислень доступні лише для читання. Поле не може бути індексованим і ключовим.

1.4. Властивості полів в базі даних Access

До основних властивостей полів відносяться обов'язкові властивості *ім'я* і *тип* поля. Додаткові властивості – необов'язкові, або відсутні, або встановлюються за замовчанням. Крім того, у полів різних типів додаткові властивості різні.

Контроль за введенням даних здійснюється на рівні поля, тому існує багато властивостей по яких йде перевірка даних.

✍ Властивості полів встановлюються при створенні таблиці або редагуються при модифікації таблиці в режимі *Таблиця/Работа с таблицами/Поля/Свойства* або в режимі *Конструктор*. Всі установки виконуються до заповнення БД записами, інакше при зміні типу полів, розміру та інших властивостей дані можуть бути загублені.

Основні властивості:

Ім'я поля – вводиться користувачем з клавіатури. Вимоги до імені поля:

– ім'я поля унікальне для кожного поля, не може бути однакових імен полів у таблиці

– довжина імені не більше 64 символів, рекомендується не більше 15 (в цьому випадку ім'я повністю поміщається у комірці таблиці, що має ширину за замовчанням

– ім'я поля містить букви, цифри і знаки підкреслення, може містити в середині пропуски і деякі знаки пунктуації (хоча останнє не рекомендується).

✍ не допускаються в імені поля: крапки (.), знаки окликів (!), квадратні дужки ([ ]), оскільки ці символи використовуються для запису виразів Access

– ім'я поля має відображати зміст інформації, що зберігається в полі

– в іменах використовують символи верхнього і нижнього регістра для легкості читання.

✎ Приклади імен полів: №СтудБил, Код_Студ, Фамилия, ОС1, ОС2.

Тип поля – обов'язкова властивість, вибирається користувачем із списку, що розкривається. За замовчанням буде вказаний тип «текстовий». Типи полів детально розглянуті у пункті 1.3.

✍ В режимі перегляду по розташуванню даних можна зорієнтуватися в типах полів: текстові вирівняні за замовчанням по лівому краю, числові по правому. Помилки у виборі типу полів принципові, і впливають на створення інших об'єктів, зв’язування таблиць, обчислення по формулах тощо. Будьте уважні, щоб не довелося переробляти все із самого початку.

Додаткові властивості (у різних типів полів розрізняються):

Опис – введений текст з'являтиметься як підказка в рядку стану при роботі з даним полем в режимі таблиці, довжина обмежена рядком стану і приблизно 100 символів.

Розмір поля – визначає діапазон даних, що вводяться.

Формат поля – визначає зовнішній вигляд для виведення даних на екран.

Число десяткових знаків – точність, задається тільки для числових полів.

Маска введення – шаблон для введення даних, спрощує введення дат, номерів телефонів тощо.

Підпис – виводиться в заголовках таблиці для зручності користувача, немає обмежень як для імені поля. Якщо не вказаний підпис, заголовки співпадають з ім'ям поля.

Значення за замовчанням – автоматично вводиться для кожного запису, що додається. Зручно при значеннях, що повторюються, або однакових даних.

Правило перевірки – вираз, який містить умову для перевірки даних. Умова перевіряється кожного разу при введенні або редагуванні даних.

Повідомлення про помилку – повідомлення, що видається на екран у випадку не виконання умови на значення.

Обов'язкове поле **(*ДА/НЕТ*)** – якщо вказано *ДА*, то користувач не має право залишати це поле не заповненим, ключові поля є обов'язковими за замовчанням.

Індексоване поле – із списку, що розкривається, користувач може вибрати один з трьох варіантів: *НЕТ, ДА(допускаються повторення),*

ДА(повторення не допускаються). За замовчанням встановлений варіант *НЕТ*, для ключових полів варіант *ДА(повторення не допускаються)*. Індексування полів прискорює пошук даних по цих полях, але затримує введення і редагування даних, оскільки здійснюється автоматична переіндексація.

Поле зі списком – об'єднує поле введення і список, що розкривається для вибору значень, прискорює введення даних. Створюється за допомогою Майстра підстановок, або зміною властивостей поля в режимі Конструктора. Стовпець підстановки може містити фіксовані значення або використовувати значення з таблиці або запиту. Майстер підстановок викликається в режимі Конструктора через типи полів, або через контекстне меню стовпця таблиці в режимі додавання поля *Щелкните для добавления / Подстановка и отношение*, або на стрічці *РАБОТА С ТАБЛИЦАМИ / ПОЛЯ / Другие поля / Подстановка и отношение*.

Ключове поле – задається користувачем в режимі Конструктора шляхом виділення поля і використання кнопки на панелі інструментів або через контекстне меню. Може бути виділено одне поле або декілька полів якщо ключ складний. Ключове поле ідентифікує записи, в ньому немає значень, що повторюються, воно завжди індексоване, по ньому сортуються записи в таблиці за замовчанням.

При спробі збереженні таблиці за відсутності ключового поля з'являється повідомлення про додавання ключового полю *Код записи* типу лічильник, але можна відмовитися від цього і задати ключ пізніше.

✍ Рекомендується задати ключове поле відразу при створенні таблиці, тоді при введенні даних здійснюватиметься перевірка по цьому полю на значення, що повторюються.

1.5. Вирази в Microsoft Access

На використанні виразів ґрунтується створення обчислюваних і підсумкових полів, введення умов для перевірки даних. Вирази використовують при створенні структури таблиці, при фільтрації даних таблиць, при створенні запитів, а також форм, звітів та макросів. Будь який вираз – це послідовність операндів та операторів. ***Оператори*** вказують, що необхідно виконати над одним або декількома операндами. ***Операндами*** можуть бути поля та їх значення, змінні, константи, літерали.

Літералом називають значення в явному вигляді, яке введено у вираз, це може бути число, рядок символів або дата.

✍ <u>Правила створення виразів</u>:

Числа вказують звичайним чином, для відокремлення дробової частини числа використовують кому, або крапку в залежності від налаштувань операційної системи комп'ютеру,

✎ наприклад, 100; 56,23; 10,00.

Дати необхідно заключати між символами (#),

наприклад, #01.02.2009#.

Символьні величини необхідно заключати у подвійні лапки ("),

наприклад, "Іванов", «пропуски» теж треба вказувати у лапках “ “.

Відмітимо, що в Access автоматизоване визначення типів даних у виразах та додавання потрібних символів(# ") у бланках запитів та фільтрів.

Константа представляє значення, що не змінюється. *True, Yes, No, False, Null* – це константи, які одразу визначені в Access. Користувач також має можливість визначити власні константи у мови VBA. Константи мають наступні значення: Yes, True – істина; No, False – брехня; Null – значення відсутнє, не визначено.

Круглі дужки у виразах використовують для зміни пріоритету операцій. Імена полів вказують у квадратних дужках (обов'язково, якщо в імені присутні пропуски).

Двокрапку (**:**) використовують як символ присвоєння.

Символи верхнього та нижнього регістру в назвах операторів не відрізняються.

Наприклад, *Сума:[Ціна]*[Кількість]; СрБал:(Ос1+Ос2+Ос3)/3*

Серед операторів виділяють наступні категорії:

1. математичні
2. рядкові
3. порівняння
4. логічні
5. допоміжні

Математичні операторі призначені для роботи з числами будь-якого типу, результат операції – число. Прості арифметичні дії *скласти*, *відняти* використовують при операціях з датами.

Список математичних операторів за убуванням пріоритету:

1. зведення в ступінь ^
2. множення *****
3. ділення /
4. ділення без остачі \
5. ділення по модулю **Mod**
6. складання +
7. віднімання –
8. мінус унарний –

Пояснимо деякі менш вживані**:**

Ділення без остачі – кожен операнд округляється до цілого, ділиться, результат знову округляється до цілого. Округлення ведеться за правилом «більш ніж 0,5». Наприклад, 100,9\6,6=14 оскільки при округленні виходить

101 поділити на 7. Цей результат відрізняється від звичайного ділення: 100,9/6,6=15,277.

✐ *Ділення по модулю* – результат це залишок, остача від ділення першого операнду на другій, наприклад: 13 Mod 4 рівне 1.

Мінус унарний – змінює знак операнду на протилежний.

Рядкові операторі призначені для роботи з символьними величинами, результат операції – символьна величина. З рядкових операторів використовують оператор *конкатенації* **&,** призначений для злиття двох рядкових величин або виразів. Оператор виконує наступні дії: у будь-якому виразі з оператором & всі типи даних будуть спочатку автоматично перетворені в рядкові, при злитті всі «хвостові» пропуски автоматично відкидаються.

✐ Приклади використання конкатенації для злиття значень полів: [Ім'я]&” “ &[Прізвище]; [Вулиця]&” “&[Номер дому]

Якщо дані не визначені й у виразі зустрічається літерал Null, діє правило: Null & Null результат Null; [Поле] & Null результат Поле

Оператори ***порівняння*** порівнюють дві величини однакового типу, результат дії оператора завжди логічна величина: Істина (True), Брехня (False), або Нуль (Null). Значення Null виходить, якщо хоч би одна з порівнюваних величин є Null, тобто значення відсутнє. Операторів порівняння 6, у порядку убування пріоритету:

Рівно	=
Не рівно	< >
Менше	<
Більше	>
Менше рівно	<=
Більше рівно	>=

✐ Приклади використання операторів порівняння для запису виразів на мові SQL: *[Вес]<10,* *[Ос1]<=5* – для числових полів

[Дата призначення]=#11.09.02# – для полів типу дата/час

[Прізвище]=”Іванов“ – для полів текстових

Логічні оператори, використовують для комбінування двох і більш операцій порівняння та отримання єдиного результату. Таким чином величини над якими виконується операція – логічні, результат теж логічна величина. Логічних операторів 6, у порядку убування пріоритету:

Логічне заперечення	**Not**	**(НЕ)**
Логічне множення (кон'юнкція)	**And**	**(І)**
Логічне додавання (диз'юнкція)	**Or**	**(АБО)**
«*Исключающееся или*»	**Xor**	
Еквівалентність	**Eqv**	
Імплікація	**Imp**	

✎ Приклади для запису умов в бланках фільтру або Запиту:
Not "США" And Not "Канада" – значення поточного поля не рівне США і не рівно Канада
"*США" Or "Канада"* – значення поточного поля рівне хоч би одному з вказаних значень США або Канада

✎ Приклад для запису виразів SQL:
[Місто] = "Київ" And [Індекс] = "252001" – вираз істинний, якщо обидва вирази істинні.

Допоміжні оператори включені в Access для сумісності з SQL. Розглянемо чотири з них:

1. *<Сим.вираз> Like <Шаблон>*. Оператор ***Like*** – порівнює рядок символів з шаблоном, результат дії – логічна величина.

Шаблон складається з символів підстановки і спеціальних символів:
***** – замінює будь-яку кількість символів
? – замінює один символ
– замінює одну цифру
[список символів] – в шаблон підставляється будь-який символ із списку
[!список символів] – в шаблон підставляється будь-який символ окрім тих, що вказані в списку.

Якщо спеціальні символи зустрічаються в порівнюваних значеннях, їх беруть в квадратні дужки.

✎ Приклади: *[Прізвище] Like "І[Вв]*"*
Like S Or Like V**
*"# 10-й круг" Like "[#]*круг"*

2. *<Вираз> Between <Значення 1> And <Значення 2>*

Оператор ***Between And*** визначає чи знаходиться величина, або результат виразу в заданому діапазоні значень. Використовують для чисел, дат, неперервних діапазонів, результат дії – логічна величина. Необхідно вказати вираз числового типу і значення меж інтервалу включно. Після обчислення виразу його результат порівнюється з граничними значеннями, якщо він потрапляє у вказаний інтервал, то весь вираз істинний.

✎ Приклади: *[Ос1] Between 3 And 5*
[Дата виконання] Between #01.04.99# And #07.04.99#

3. *<Вираз> In (Значення1, Значення2, Значення3, …ЗначенняN)*

Оператор ***In*** визначає чи знаходиться величина, або результат виразу серед вказаних в списку. Використовують для будь-яких типів полів, дискретних значень**.** Список значень указують в дужках, через кому. Після обчислення виразу його результат порівнюється з вказаними в списку, весь вираз істинний, якщо є хоч би один збіг.

✎ Приклади: *[Ос1] In (4,5)*
[Прізвище] In ("Іванов", "Петров", "Сидоров")

4.Оператори ***Is Null, Is Not Null,*** перевіряють чи відсутнє значення. Використовують для будь-яких типів полів.

Приклад: *[Дата виконання] Is Null* – істинно, якщо дата відсутня,
[Дата виконання] Is Not Null – істинно, якщо дата присутня.

Оператори ***ідентифікації*** використовують у виразах SQL для посилань на об'єкти та їх властивості:
Знак (**!**), оклику, – для розділення посилань на об'єкти та елементи
Крапка (**.**) – для вказівки значення поля або властивості елементів.

Повний ідентифікатор необхідно використовувати для посилання на елемент управління, розташований не в поточній формі або звіті.

Приклади: Клієнти! [Назва] – Ім'я таблиці![Ім'я поля]
Клієнти.[Прізвище] – Ім'я таблиці.[Значення поля]
Forms![Особисті справи] – Тип об'єкта![Ім'я об'єкта]
Reports![Рахунок]![НазваніЄПолучателя].Visible –
Тип об'єкта![Ім'я об'єкта![Елемент управління].Вивід на екран

Розглянемо детальніше останній приклад: в Access автоматично створюються сімейства Forms і Reports для кожної бази даних. Сімейство Forms складається зі всіх відкритих форм, а сімейство Reports зі всіх відкритих звітів. Таким чином перше посилання Reports на тип об'єкта. Компонент [Рахунок] визначає посилання на звіт з ім'ям «Рахунок» у сімействі Reports. Компонент [НазваніЄПолучателя] визначає посилання на елемент управління «НазваніЄПолучателя» в звіті «Рахунок». Компонент «Visible» визначає посилання на властивість Вивід на екран (Visible) елементу управління «НазваніЄПолучателя».

Будівник виразів. Для створення виразів зручно використовувати вбудований Майстер-будівник виразів, для виклику якого слід клацнути кнопку *Построитель* на панелі інструментів в режимі Конструктора (рис. 1).

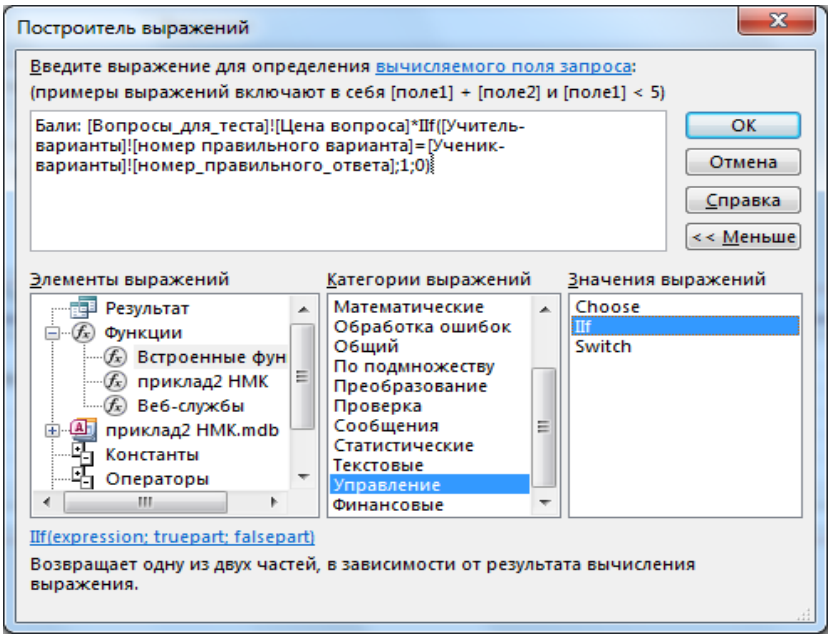


Рис.1. Будівник виразів

У верхній частині вікна будівника розташовано поле виразу. Допускається безпосереднє введення виразу в полі виразу. Нижче знаходиться допоміжний розділ, призначений для створення елементів виразу та їх подальшої вставки у вираз. У нижній частині вікна будівника знаходяться три поля:

- У лівому полі *Элементы выражений* виводяться папки, що містять таблиці, запити, форми, об'єкти бази даних, вбудовані і визначені користувачем функції, константи, оператори і загальні вирази.
- У середньому полі *Категории выражений* задаються певні елементи або типи елементів для папки, заданої в лівому полі. Наприклад, якщо вибрати в лівому полі *Вбудовані функції*, то в середньому полі з'явиться список всіх типів функцій Microsoft Access.
- У правому полі *Значения выражений* виводиться список значень (якщо вони існують) для елементів, заданих лівим і середнім полями. Наприклад, якщо вибрати в лівому полі *Вбудовані функції* і тип функції в середньому, то в правому полі буде виведений список всіх вбудованих функцій вибраного типу.
- У рядку стану виводиться ім'я вибраної функції і пояснення до неї.

Вибравши в нижньому полі будівника необхідну функцію або ім'я поля двічі клацніть по ньому – потрібний елемент вставиться в поточну позицію у верхньому полі будівника виразів

Видалення елементів – клавіша *Del, Backspase*.

✍ Щоб нові імена полів з'явилися в будівнику виразів, слід заздалегідь зберегти таблицю або запит, що містять такі поля.

Якщо функція або об'єкт не виводяться в нижній частині будівника виразів, то це значить, що їх використання неприпустимо в позиції, з якої був викликаний будівник. Наприклад, папки *Таблиці, Запити, Форми і Звіти* не доступні при запуску будівника виразів для властивості поля *Умова на значення* в режимі конструктора таблиці.

1.6. Функції у Microsoft Access

Крім імен полів, операторів, літералів у виразах використовуються функції. Функція повертає значення, яке є результатом розрахунків або виконання інших операцій. В Access існує значна кількість вбудованих функцій, близько 140 шт. У мові Visual Basic користувач має можливість створювати власні функції. Вбудовані функції дещо співпадають з функціями в мові програмування VBA і в електронних таблицях Excel. Щоб викликати функцію треба задати її ім'я та в дужках вказати аргументи функції, тобто вхідні значення, результат розрахунків або інших дій підставляється у вираз.

Залежно від типів даних, які опрацьовуються, функції розділені на наступні категорії, основні з яких: математичні, текстові, дата/час, перетворення типу, статистичні.

Приклади функцій за категоріями наведено у табл. 3-6.

Таблиця 3.

Математичні функції, для роботи з числовими даними:

Функція	Опис дій	Приклад	Значення, що повертається
Fix( )	повертає цілу частину числа	Fix(13,5) Fix(-13,5)	13 -13
Int( )	повертає цілу частину числа з округленням	Int(13,5) Int(-13,5)	13 -14
Sqr( )	повертає корінь квадратний від числа	Sqr(144)	12
Sin( )	повертає синус кута	Sin(pi/4)	0,707106842423
Cos( )	повертає косинус кута	Cos(pi/4)	0,707106719949
Tan( )	повертає тангенс кута	Tan(pi/4)	1,000000173211
Atn( )	повертає арктангенс кута в радіанах	Atn(1)	0,7853982
Log( )	повертає логарифм натуральный числа	Log(10)	2,302585
Exp( )	повертає експоненту, тобто ***e*** у степені	Exp(2,302585)	9,9999990700
Abs( )	повертає абсолютне значення числа	Abs(-1234,5)	1234,5
Sgn( )	повертає знак числа	Sgn(-13,5)	-1
Rnd( )	повертає будь яке випадкове число	Rnd	1,653125E-02

Таблиця 4.

Текстові функції, для роботи з символьними даними:

Функція	Опис дій	Приклад	Значення, що повертається
Left(,)	вирізає з рядкової величини N символів зліва	Left(”ABCDEF”,3)	ABC
Right(,)	вирізає з рядкової величини N символів справа	Right(”ABCDEF”,3)	DEF
Len( )	повертає довжину рядка	Len(”ABCDE”)	5
Space( )	вставка вказаного числа пропусків	Space(5)	
Mid(, ,)	вирізає декілька символів з певної позиції	Mid(”ABCDE”,2,3)	BCD

Trim( )	видаляє пропуски зліва та справа	Trim(” ABC ”)	ABC
Ltrim( )	видаляє пропуски зліва	Ltrim(” ABC”)	ABC
Rtrim( )	видаляє пропуски справа	Rtrim(”ABC ”)	ABC
InStr( )	повертає позицію входження строки символів в іншу строку	InStr(”ABCD”,”C”)	3
Lcase( )	перетворює у нижній регістр	Lcase(”ABCDE”)	abcde
Ucase( )	перетворює у верхній регістр	Ucase(“abcde”)	ABCDE
String( )	повторює строку вказаної довжини	String(5,”A”)	AAAAA
StrComp( )	порівнює строки	StrComp(“ABC”,”abc”)	0

Таблиця 5.

Функції типу дата/час, для роботи з даними типу дата/час:

Функція	Опис дій	Приклад	Значення, що повертається
Date( )	повертає системну дату	Date( )	4.06.09
Now( )	повертає системну дату і час	Now( )	4.06.09 15:21:47
Time( )	повертає системний час	Time( )	15:21:47
Year( )	повертає рік в чотири цифри	Year(#3.06.09#)	2009
Month( )	повертає місяць	Month(#3.06.09#))	6
Weekday( )	повертає номер дня тижня, рахує з неділі	Weekday(#3.06.09#)	4
Day( )	повертає число місяця	Day(#3.06.09#)	3

Таблиця 6.

Функції перетворення типу, виконують перетворення одних типів даних в інші:

Функція	Опис дій	Приклад	Значення, що повертається
Str ( )	число перетворить в рядок	Str (123,45)	123,45

Val( )	рядок перетворить в число	Val(”123,45”)	123,45
Format( )	перетворить дані до формату користувача	Format(Date,”dd-mmm-yy”)	26-фев-98
Asc( )	повертає код ASCII першого символу строки	Asc(“C”)	67
Chr( )	повертає символ який відповідає коду	Chr(67)	C
DataSerial( )	повертає дату, якщо вказати рік, місяць, день	DataSerial(95,7,15)	15.07.95
TimeSerial( )	повертає час, якщо вказати години, мінути, секунди	TimeSerial(11,57,20)	11:57:20
TimeValue( )	рядок перетворить в час	TimeValue(”11:57”)	11:57:00

Статистичні функції виконують групові операції над набором даних, які містяться в полі форми, звіту або запиту.

Avg( ) середнє значення від числових значень в полі
Sum( ) сума числових значень в полі
Max( ) максимальне з числових значень в полі
Min( ) мінімальне з числових значень в полі
Var( ) дисперсія розподілу числових значень в полі
StDev( ) стандартне відхилення числових значень в полі
Count( ) підрахунок кількості не порожніх значень в поле, будь-який тип поля
First( ) перший зі списку, будь-який тип поля
Last( ) останній зі списку, будь-який тип поля

Додаткові функції загального призначення використовують у формах, звітах, макросах та запитах для перевірки значень.

✎ Наприклад: *IIF([Объем_заказа]Mod12=0,”дюжина”,”штук”)* повертає значення *дюжина,* якщо заказано 12 одиниць товару, у противному випадку повертає значення *штук.*

1.7. Особливості зберігання та обробки інформації в базах даних

Для БД пред'являються особливі вимоги з погляду безпеки, а інформація в БД має велику цінність. Одна з найважливіших функцій СУБД є підтримка цілісності БД, що дозволяє забезпечити використання правильної, заздалегідь перевіреної інформації різними користувачами. Тому робота з файлами БД відрізняється від роботи з файлами інших програм обробки даних.

Для збереження інформації використовується комбінований підхід: звичайний для загальних операцій збереження та спеціальний для даних, характерний лише для СУБД.

Операції зміни структури БД, тобто створення або модифікація базових таблиць та інших об'єктів, відбуваються за командами користувача. В цих операціях по збереженню бере участь операційна система, з'являється повідомлення *Сохранить? ДА, НЕТ, ОТМЕНА*. При цьому зберігається файл БД. Таки операції ніколи не проводять з БД, що знаходиться в комерційній експлуатації, а тільки з її копією.

✍ Операції збереження зміни даних, що не змінюють структуру БД, автоматизовані і виконуються без попередження, відбуваються в обхід операційної системи. Якщо Ви змінюєте дані в таблиці, зміни зберігаються негайно і автоматично, після переходу до наступного запису. Не можна відмінити операцію!

СУБД працює при операціях зміни даних безпосередньо з електронним носієм (диском), при цьому здійснюються так звані транзакції. ***Транзакція*** – це декілька узгоджених операцій по зміні змісту БД. Системний компонент – ***адміністратор транзакцій*** забезпечує оновлення БД так, щоб БД після зміни знов опинилася в узгодженому стані. Якщо відбувся збій, транзакція відміняється, початкові дані відновлюються з резервного файлу на диску. Тому навіть якщо «закрити БД без зміни» дані будуть внесені. Таким чином СУБД «захищає» інформацію від збоїв електропостачання, від суб'єктивного чинника – забудькуватості оператора, тощо.

Такий підхід щодо збереження інформації призводить до незвичного виконання вже знайомих користувачеві операцій з файлом даних. Якщо відкрити наступну БД, попередня автоматично закривається. Якщо треба створити нову БД, спочатку треба створити файл БД вказуючи ім'я файлу та папку розташування. Кожен об'єкт у БД створюється та зберігається окремо починаючи з базових таблиць. Команда *Сохранить как*…діє на конкретний об'єкт у БД. При закриванні таблиць, форм, запитів відмінити зміни змісту неможливо!

Розділ 2. ПРАКТИЧНІ ОСНОВИ РОБОТИ СУБД ACCESS

Початок роботи з Microsoft Access

При запуску програми Access з'являється допоміжне вікно для створення нової БД на основі шаблону або відкриття вже існуючої БД. Під відкриттям БД розуміють підключення файлу БД до робочої області оперативної пам’яті. Для відкриття БД використовують стандартний підхід з використанням меню «*Файл / Открыть*», також список БД які недавно відкривалися. Файл БД можна відкрити безпосередньо з папки або з Робочого столу, в цьому випадку програма Access запускається і відразу відкриває БД.

Існує чотири режими роботи Access.

1. Початковий режим – немає відкритих БД, можуть виконуватися файлові операції: стискати, конвертувати, зашифровувати, розшифровувати, відновлювати пошкоджені дані. Для цього використовують пункти меню: *РАБОТА С БАЗАМИ ДАННЫХ / Сервис / Сжать или восстановить базу данных*
2. Режим виконання або перегляду (основний режим) у відкритої БД для перегляду і редагування даних. Можливо додавання записів, пошук, сортування, фільтрація записів в режимі таблиці, запиту або форми (рис. 2).
3. Режим Конструктора у відкритої БД для створення і зміни структури таблиць, форм, звітів, запитів. Викликається за допомогою кнопки на панелі інструментів, або через контекстне меню об’єкту.
4. Режим Макету для редагування форм та звітів. Викликається за допомогою кнопки на панелі інструментів, або через контекстне меню об’єкту.

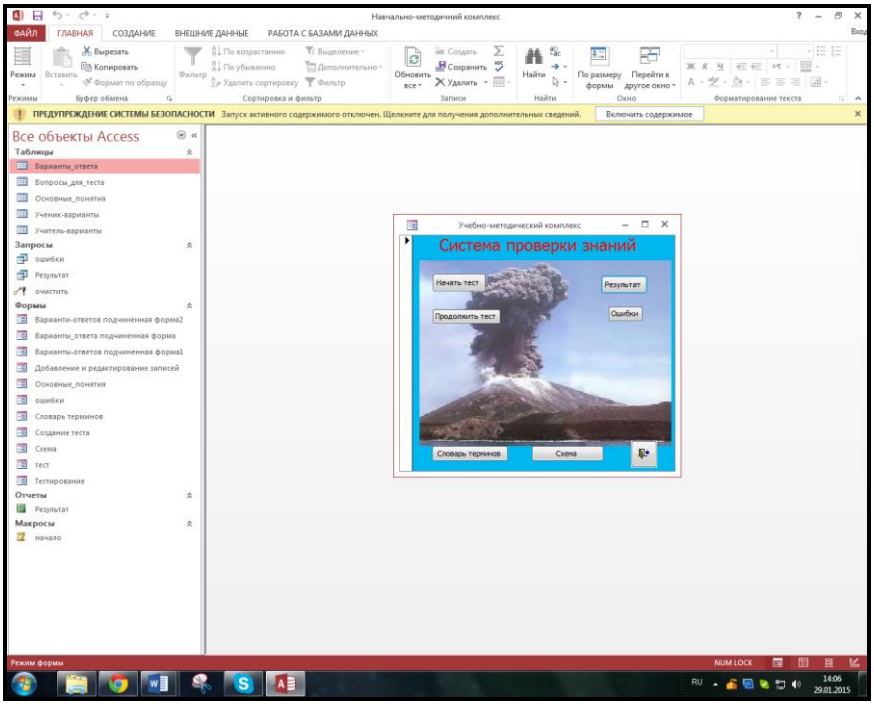


Рис. 2. Робоче вікно основного режиму роботи Access

Основне вікно програми містить наступні елементи:

1. Рядок Заголовка
2. Рядок меню: *ФАЙЛ, ГЛАВНАЯ, СОЗДАНИЕ, ВНЕШНИЕ ДАННЫЕ, РАБОТА С БАЗАМИ ДАННЫХ*
3. Стрічку (панель) інструментів
4. Вікно БД
5. Рядок стану

Вікно БД складається з двох частин:

- основна частина вікна призначена для роботи з об’єктами БД, яких можна відкрити декілька у будь якому режимі
- вертикальний ряд зліва містить вкладку *Все объекты Access.*

Дії над об'єктами виконують за допомогою контекстного меню і кнопок на панелі інструментів, щоб відкрити об'єкт виконайте подвійне клацання миші. Для більш зручної роботи бажано вказати параметри вкладки: *Тип объекта, Все объекты Access* (Рис. 3).

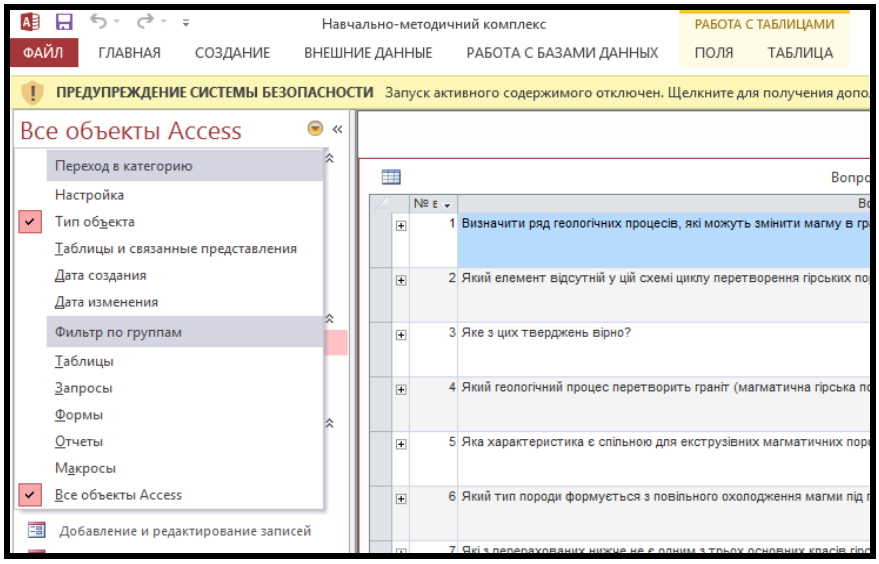


Рис. 3. Налагодження вкладки обєктів Access

Базова таблиця в режимі перегляду має рядок навігації унизу для переміщення по записах та спеціальні маркери зліва.

Активна запис указується позиційним курсором (прямокутною рамкою). Це запис в якій знаходиться символьний курсор.

Редагований запис указується олівчиком ✐. Доти поки запис є редагованим всі зміни можна відмінити за допомогою клавіші Esc.

Останній вільний рядок помічено символом «зірочка» *****, він призначений для додавання нового запису.

Якщо зліва від записів є позначка **+** , ця таблиця зв’язана з іншою таблицею.

Навігація по таблиці, тобто перехід від запису до запису здійснюється за допомогою миші, або клавіатури (клавіш управління курсором) або за допомогою кнопок навігації, розташованих в нижній частині вікна БД. У рядку навігації указується порядковий номер активного запису і загальна кількість записів в таблиці а також присутність фільтру.

Запити, форми у режимі перегляду мають аналогічні елементи керування.

2.1. Створення бази даних Access

Запустити Access. Вибрати «*Пустая база данных рабочего стола*». Вказати ім'я нової БД в діалоговому вікні і папку для збереження файлу. За замовчанням ім'я базі даних – *База данных1.accdb* або *База данных2.accdb*, якщо така вже існує. Файли зберігаються в папку «*Документы*». Знов створений файл займає 340 Кб для Access 2013 (порівняйте з 60 Кб для Access 2003). Перейменувати файл можливо буде тільки засобами операційної системи, закривши програму Access.

За відсутності допоміжного вікна користуються пунктом меню *Файл/Создать*, далі аналогічно.

Якщо серед шаблонів БД є такі, що підходять, можна завантажити готові БД зі створеною структурою, та заповнити їх даними.

Починати створення структури БД необхідно зі створення базових таблиць. Для створення нової таблиці існує декілька способів.

– У вікні програми на стрічці *СОЗДАНИЕ* вибрати *Таблица*
– У вікні програми на стрічці *СОЗДАНИЕ* вибрати *Конструктор таблиц*
– На стрічці *СОЗДАНИЕ/ Таблицы* можна використати списки *Share Point* для імпортування структури таблиці з шаблону.

✍ За замовчанням таблиці пропонується ім'я *Таблица1, Таблица2*…. Рекомендується при збереженні структури дати ім'я таблиці, так щоб воно відображало зміст таблиці, при необхідності таблицю можна перейменувати через контекстне меню таблиці.

Режим таблиці, або його ще називають ручний режим введення даних. Користувач доповнює та заповнює звичайну таблицю у вікні БД (рис. 4). За замовчанням поля в ній мають назви Код, Поле1, Поле2, Поле3… які краще відразу перейменувати так, щоб заголовки таблиці відповідали змісту полів. Зробити це можливо й потім через контекстне меню поля. Щоб додати нове поле клацніть *Щелкните для добавления* та оберіть потрібний тип поля зі списку. Команди на стрічці *РАБОТА С ТАБЛИЦАМИ / ПОЛЯ /...* призначені для встановлення властивостей полів. Є можливість вставити або видалити поле через контекстне меню.

Таблицю заповнюють по рядках, в межах одного стовпця дані одного типу, перехід від стовпця до стовпця – клавіша Tab, або Enter або клацання мишею. Після введення даних, можна перейти в режим Конструктора і відредагувати структуру таблиці. Перед цим необхідно зберегти дані і ввести ім'я нової таблиці.

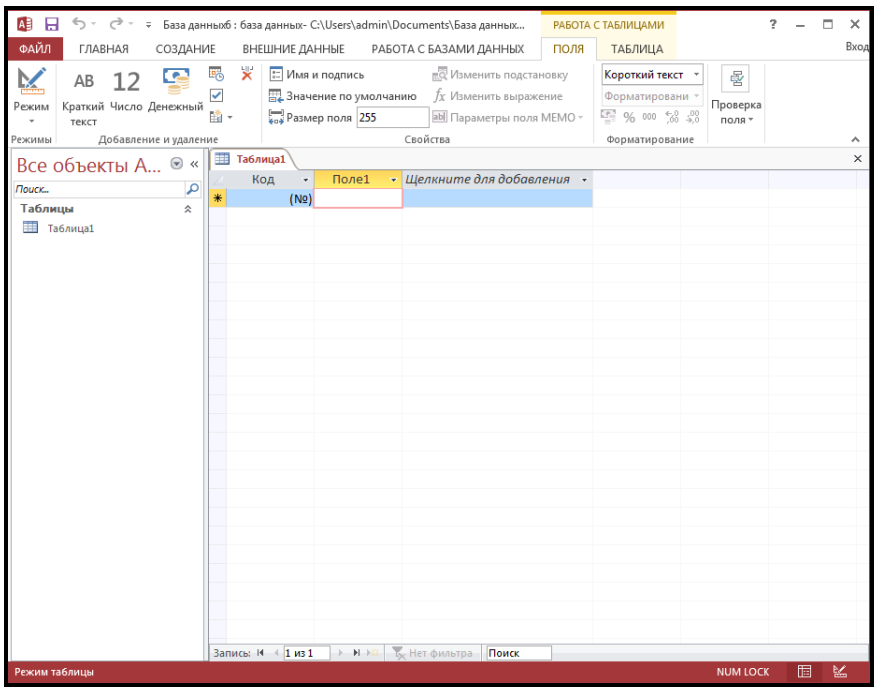


Рис. 4. Створення таблиці

Конструктор таблиць – найгнучкіший та оптимальний засіб для створення структури. У режимі Конструктору дані таблиці не відображаються, лише структура, тобто список полів. Користувач визначає самостійно всі властивості полів в таблиці. Верхня область вікна Конструктора (рис. 5) призначена для опису основних властивостей полів, нижня область – для опису додаткових. Справа внизу область контекстно-залежної довідки. Панель інструментів відповідає роботі в режимі Конструктора: видалити рядки, додати рядки, задати ключове поле, тощо. При збереженні таблиці треба задати ім’я. У режимі Конструктора також модифікують структуру таблиці.

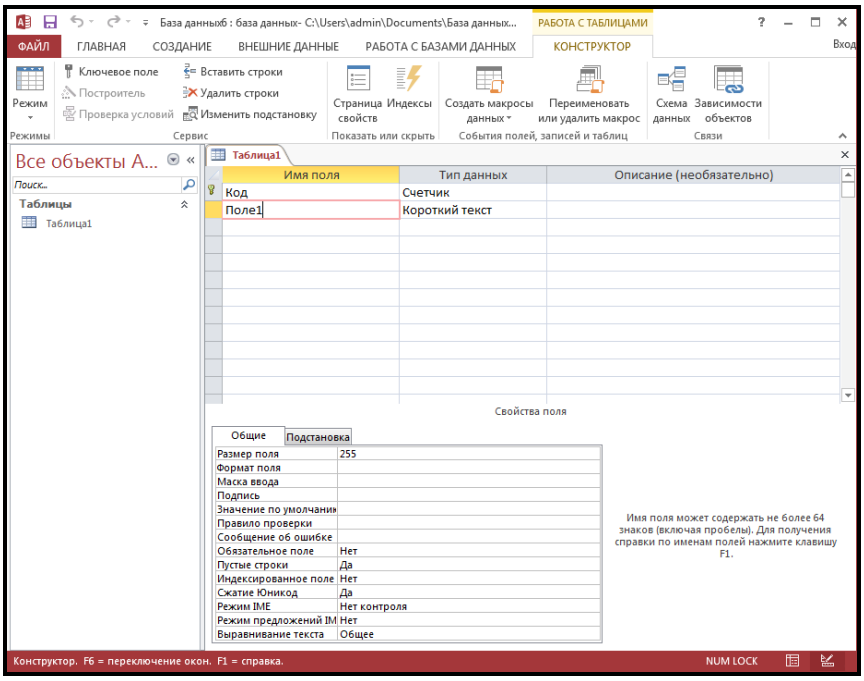


Рис. 5. Таблиця у режимі Конструктора

Введення і редагування даних таблиць. Введення даних зручніше здійснювати за допомогою форми, але для тестування створеної таблиці треба ввести пробні дані в режимі таблиці.

Переміщення по полях таблиці: клавіші управління курсором, клавіша Tab, маніпулятор миша.

Відмова від введеного значення – Esc, відмова від введеного запису – двічі Esc. Відмінити введені дані можна поки запис в режимі редагування (відмічена олівчиком), при переході до іншого запису автоматично здійснюється збереження даних на диску.

Для ***видалення записів таблиці***, виділити одну або декілька записів і натиснути клавішу Delete.

✍ Після підтвердження видалення записів відновити їх неможливо, інформація є загубленою! Видалити можна або весь запис цілком (декілька штук), або все поле цілком (декілька штук), або дані з однієї комірки таблиці. Видалити прямокутний блок записів з середини таблиці не можна, при цьому порушуються умови цілісності даних.

Створення міжтабличних зв'язків. Зазвичай в БД організовано декілька взаємозв'язаних таблиць, в цьому випадку створюють таблиці, тестують їх ввівши декілька записів і після визначають зв'язки з іншими таблицями. Від того, як таблиця зв'язана з іншими, залежить яке з її полів слід призначити первинним ключем.

✍ Необхідно**,** щоб поля двох таблиць, по яких вони зв'язуються між собою мали однаковий тип і підтип даних (тобто розмір), імена можуть і не співпадати.

Зв'язки створюються та редагуються у вікні «Схема даних» (рис. 6). Для цього користуються контекстним меню області вікна, таблиць і ліній зв'язку, а також командами на стрічці *РОБОТА СО СВЯЗЯМИ / КОНСТРУКТОР*. Команда *Очистить макет* не видаляє зв'язки, а лише знімає їх відображення у вікні схеми даних.

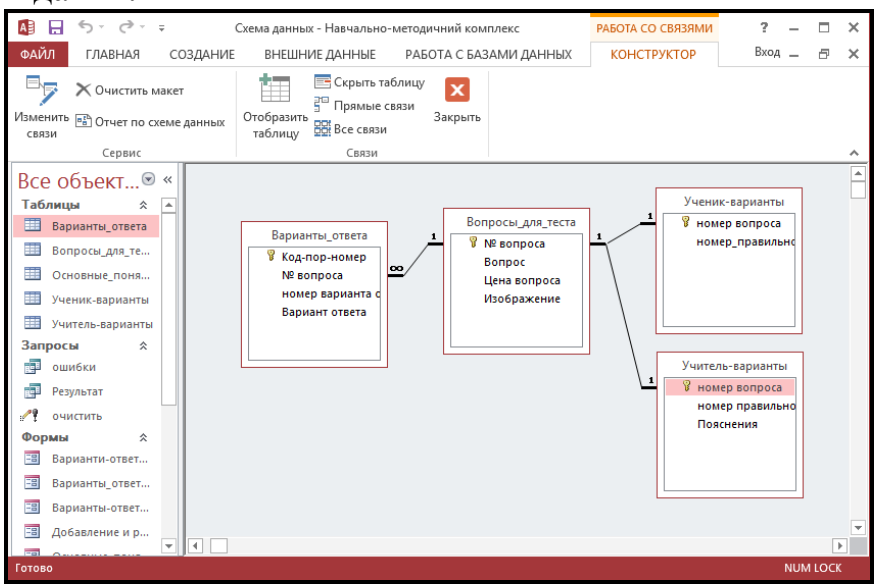


Рис. 6. Схема даних

Алгоритм створення зв'язків:

- закрити всі відкриті у вікні БД таблиці, інакше операцію може бути заблоковано
- скористатися кнопкою «Схема даних» на панелі інструментів
- додати таблиці у вікно схеми даних
- для створення зв’язку слід за допомогою миші перетягнути поле зв’язку з однієї таблиці в іншу на відповідне поле, причому дуже важливо з якої в яку перетягуємо, необхідно з головної таблиці у підлеглу
- у вікні зв'язку переконатися в правильності створеного відношення, вибрати тип об'єднання і підключити параметр *Обеспечение целостности данных*

Тип об'єднання задає спосіб перегляду зв'язаних записів на основі заданого відношення. Для перегляду даних одночасно двох таблиць, наприклад в запиті, вибирають один з трьох типів об'єднання:

рівне **або** ***внутрішнє*** об'єднання, об'єднання тільки тих записів, в яких зв'язані поля обох таблиць співпадають, при цьому записи без пари («вдови» с головної таблиці і «сироти» з підлеглої) не відображаються, це найбільш поширений тип об'єднання, встановлений за замовчанням;

зовнішнє праве об'єднання, об'єднання ВСІХ записів з першої таблиці і лише тих записів з другої таблиці, в яких зв'язані поля співпадають («вдови» відображаються, «сироти» ні), використовують для зручного додавання записів у підлеглій таблиці;

зовнішнє ліве об'єднання, об'єднання ВСІХ записів з другої таблиці і лише тих записів з першої таблиці, в яких зв'язані поля співпадають («сироти» відображаються, «вдови» ні), використовують для пошуку записів-«сиріт» та виправлення помилок, при наявності записів-«сиріт» створення зв’язку з забезпеченням цілісності даних неможливе.

✍ Підключення параметру ***забезпечення цілісності даних*** є обов’язковим для робочих БД, воно дозволяє уникнути записів-«сиріт» і неправильно зв'язаних записів. При збереженні нового або зміненого запису автоматично перевіряються значення критичних полів. Якщо спробувати в підлеглу таблицю внести дані про об'єкт не зв'язаний з головної таблицею, транзакцію буде відмінено, в збереженні даних відмовлено. Тобто дані треба вносити спочатку у головну таблицю, а потім у підлеглу. Забезпечення цілісності даних не вдається підключити, якщо цілісність даних порушена, в цьому випадку необхідно змінити дані.

Для підтримки цілісності даних можна за необхідності підключити ще два параметра: ***каскадне оновлення зв'язаних полів*** та ***каскадне видалення зв'язаних записів,*** перший автоматизує зміни значень зв’язаних полів для підлеглої таблиці, другий дозволяє автоматично вилучити зв’язані записи в підлеглій таблиці. Якщо каскадне видалення не підключене, при спробі видалення запису з головної таблиці за наявності зв'язаних з нею записів теж буде відмовлено.

Лабораторна робота №1. Створення бази даних

Мета: Придбання практичних навиків створення БД, проектування таблиць, зміни структури таблиць, установки зв'язків між таблицями.

Завдання.

У програмі Access створити нову базу даних з назвою «Деканат-…» вказуючи особисті дані (наприклад, «Деканат-Иванов411гр»). У БД «Деканат» створити таблиці з назвами: «Групи», «Студенти», «Оцінки», використовуючи різні способи для створення таблиць, та з'єднавши таблиці різними типами зв'язку.

☞*Вказівки.* Таблицю «Групи» створити за допомогою режиму Конструктора. Список полів та їх властивостей у табл. 7.

Таблиця 7

Таблиця «Групи».

Ім'я Поля	Тип даних	Опис	Властивості
Код	Лічильник	–	–
Ном Гр	Текстовий	Номер групи	Обов'язкове, ключове, розмір:3
Староста	Текстовий	–	–
Спец	Текстовий	Спеціальність	Розмір: 30, підпис: Спец-сть
Кіл_Ст	Числове		Ціле, число десятк. знаків 0, умова на значення <=30, повідомлення про помилку: «Кількість студентів у групі перевищує 30 чіл.», значення за замовчуванням 20
ПрохБал	Числове	Прохідний бал	Одинарне з плаваючою крап., число десяткових знаків 2, значення за замовчуванням 3, умова на значення <=5.

Заповніть таблицю «Групи» записами про 5 груп в режимі таблиці, тестуючи при цьому властивості полів, перевіривши всі можливі варіанти. Зверніть увагу на рядок стану та заголовки таблиці. З'ясуйте, як впливає наявність вказаних властивостей на внесення записів. При необхідності поверніться в режим конструктора і модифікуйте структуру, змініть властивості полів.

☞*Вказівки.* Таблицю «Студенти» створити за допомогою режиму Таблиці. Список полів та їх властивостей у табл. 8. Відкоригуйте структуру за допомогою Конструктора. Перетворіть у таблиці «Студенти» поле НомГр у поле зі списком груп за допомогою Майстра Підстановок, це автоматизує введення даних і позбавить від помилок при введенні інформації.

Таблиця 8

Таблиця «Студенти».

Ім'я Поля	Тип даних	Властивості
Код_Ст	Счетчик	
Ном_Гр	Текстовий	Обов'язкове, розмір: 3
НомСтбил	Текстовий	Обов'язкове, розмір 5, ключове
Прізвище	Текстовий	
Ім’я	Текстовий	
По-батькові	Текстовий	
Місто	Текстовий	
Адреса	Текстовий	
Телефон	Текстовий	
СімСтан	Логічне	
ДатаНародж	Дата/година	

Для тестування заповніть таблицю «Студенти» декількома записями.

☞*Вказівки.* Таблицю «Оцінки» створити способом який для Вас найзручніший. Відкоригувати за допомогою Конструктора. Таблиця «Оцінки» повинна містити відомості про екзаменаційну сесію: оцінки (від 2 до 5), отримані на іспитах – Экз1, Экз2, Экз3, а також відомості про здані заліки – Зач1, Зач2, Зач3. Правильно вкажіть типи полів у таблиці: числові, логічні, текстові. У режимі Конструктора модифікуйте таблицю виправивши типи і властивості полів.

☞*Вказівки.* Встановіть зв'язок між таблицями. Таблиця «Групи» повинна бути зв'язана по полю Ном_Гр із таблицею «Студенти», тип зв'язку один-до-багатьох. Таблиця «Оцінки» повинна бути зв'язана з таблицею «Студенти» по полю НомСтбил, тип зв'язку один-до-одного. Ключове поле скопіюйте через Буфер Обміну з таблиці «Студенти».

☞*Вказівки.* Заповніть таблицю «Студенти» відомостями про 15 студентів з трьох різних груп та заповніть таблицю «Оцінки» даними.

? Контрольні запитання

1. Особливості збереження інформації в БД.
2. Етапи створення БД.
3. Призначення базових таблиць.
4. Алгоритм створення таблиці за допомогою Конструктора.
5. Алгоритм створення таблиці введенням даних.
6. Типи полів та їх властивості.
7. Додавання і видалення полів у таблиці, зміна порядку полів.
8. Типі зв'язків між таблицями. Забезпечення цілісності даних.
9. Створення, редагування й видалення зв'язків між таблицями.
10. Засоби, що дозволяють контролювати й автоматизувати введення даних.

2.2. Сортування, пошук, фільтрація записів бази даних

Режим виконання або перегляду – основний режим при роботі в БД. Він призначений для перегляду і редагування даних. Можливо додавання записів, пошук, сортування, фільтрація записів в режимі таблиці, запиту або форми. Сортування і відбір записів в таблиці Access виконується аналогічно цім операціям в таблиці Excel. Тобто на заголовках таблиці у режимі перегляду одразу є кнопки фільтру, що спрощує виконання сортування та відбору записів. Однак в Access ці операції виконуються швидше, завдяки автоматичному використанню індексів.

Сортування записів – це відображення записів і стовпців таблиці в потрібному порядку. *Види сортувань:* По одному полю та По декількох полях. *Типи сортувань:* За збільшенням та За зменшенням.

При сортуванні *по одному полю* досить помістити курсор в це поле в таблиці або формі, або виділити його клацнувши на заголовку поля в таблиці. При сортуванні *по декількох полях* (два поля) необхідно виділити декілька полів поряд, утримуючи клавішу **Shift** і клацаючи по заголовках, або «протягнувши» по заголовках мишею утримуючи ліву кнопку. Виконати *По возрастанию* або *По убыванию*.

✍ Сортування по декількох полях здійснюється зліва направо, спочатку по полю розташованому ліворуч, а потім для однакових значень лівого поля по полю розташованому праворуч, сортування одночасно по декількох полях має сенс в тому випадку, якщо в першому полі є значення, що повторюються.

Для ***переміщення*** *полів* в таблиці перетягують за заголовок виділений стовпець.

Відміна сортування – виконати команду *Удалить сортировку*. Якщо закрити таблицю відмовившись від збереження змін, то відміняється сортування записів і переміщення стовпців таблиці.

Для зручнішого перегляду записів можна ***змінити ширину стовпців***, ***приховати деякі стовпці***, тобто не відображати їх при перегляді, ***закріпити стовпці*** щоб позбавитися горизонтального скролінгу. Всі ці дії виконують через контекстне меню виділених стовпців.

Пошук записів. Пошук здійснюється по поточному полю або по поточній таблиці (формі), відповідно до вказаного зразка пошуку. По індексованих полях пошук здійснюється швидше.

Послідовність дій:

1. У режимі таблиці або форми помістити курсор в полі, по якому здійснюється пошук
2. Викликати операцію пошуку – кнопка *Найти* на панелі інструментів (зображення біноклю), або комбінація клавіш Ctrl+F
3. У діалоговому вікні *Поиск* у рядку *Образец* ввести необхідне значення
4. Вказати параметри пошуку, встановивши один з декількох запропонованих варіантів
5. Клацнути по кнопці *Найти далее*

Параметри пошуку:

Збіг: Поле цілком (встановлено за замовчанням
З початку поля
З будь-якою частиною поля

Перегляд: Все (встановлено за замовчанням
Вгору
Вниз

З урахуванням регістра

З урахуванням формату полів

Якщо записи не знайдені на екрані з'являється повідомлення *Поиск записей в .... завершен. Образец не найден*. Якщо запис знайдений, курсор встановиться в полі з вказаним значенням, цей запис стане активним, можливо подальше редагування запису.

Пошук і Заміна. Для автоматичного редагування групи записів використовують операцію *Замена*. Послідовність дій аналогічна операції пошуку.

Послідовність дій:

1. У режимі таблиці або форми помістити курсор в полі, по якому здійснюється пошук
2. Викликати операцію *Найти/Поиск и замена* (комбінація Ctrl+H)
3. У діалоговому вікні *Замена* у рядку *Образец* ввести необхідне значення для пошуку
4. У рядку *Заменить на* ввести необхідне значення
5. Вказати параметри пошуку
6. Клацнути по кнопці *Заменить* або *Найти далее*, або *Заменить все*

✍ Після заміни значень відміна операції неможлива!
Операція заміни блокується якщо порушуються умови забезпечення цілісності даних.

Фільтрація записів – це відбір і відображення тих записів, що задовольняють вказаним умовам. Умови для відбору записів зберігаються в так званому «Бланку фільтру» і записуються за спеціальними правилами по різному для різних типів даних. Умови для відбору записів можна вносити одночасно або послідовно, редагувати, видаляти, почистити бланк фільтру, щоб видалити все відразу. Встановлений фільтр можна застосувати або відмовитися від фільтру, клацнув кнопку на панелі інструментів *Фильтр*. Встановлений фільтр автоматично зберігається і може бути застосований при наступних сеансах роботи з таблицею.

Існує **декілька способів** установки фільтру.

1. через кнопку фільтру у заголовку стовпця
2. через контекстне меню комірки
3. через фільтр по виділеному *Выделение* на стрічці *Главная*
4. через фільтр по формі *Дополнительно /Фильтр по форме*
5. через розширений фільтр *Дополнительно/Расширеный фильтр.*

Способи можна комбінувати, якщо не видаляти умови попередніх фільтрів, вони застосовуються водночас. Команда *Дополнительно/ Очистить все фильтры* видаляє всі умови. Для розширеного фільтру та для фільтру по формі застосовується команда *Почистить бланк*. Умови, які встановлено, краще переглядати безпосередньо на бланку фільтру в розширеному фільтру.

Фільтр по виділеному. Для використання цього способу необхідно виділити дані або частину даних в комірці. Залежно від того, що саме виділено встановлюються умови фільтру, програма пропонує декілька варіантів умов в залежності від типу даних.

Фільтр по формі дозволяє вносити умови за допомогою спеціальної форми. При цьому користуються списками, які розкриваються, математичними операторами для запису умов і ярличком «*ИЛИ*» в нижній частині вікна фільтру, якщо умови поєднуються через логічну операцію «АБО». У разі, коли умови внесені в одному рядку форми, вони об'єднуються операцією «И», тобто виконуються водночас. Після запису умов дають команду застосувати фільтр – кнопка *Фильтр* на панелі інструментів. Операція *Очистить бланк* видаляє всі записані умови.

✍ Будьте уважні, не забувайте очищати бланк фільтру, інакше попередні умови накладатимутися на нові умови.

Розширений фільтр надає більше можливостей, дозволяє вносити умови безпосередньо в бланк фільтру (рис. 7).

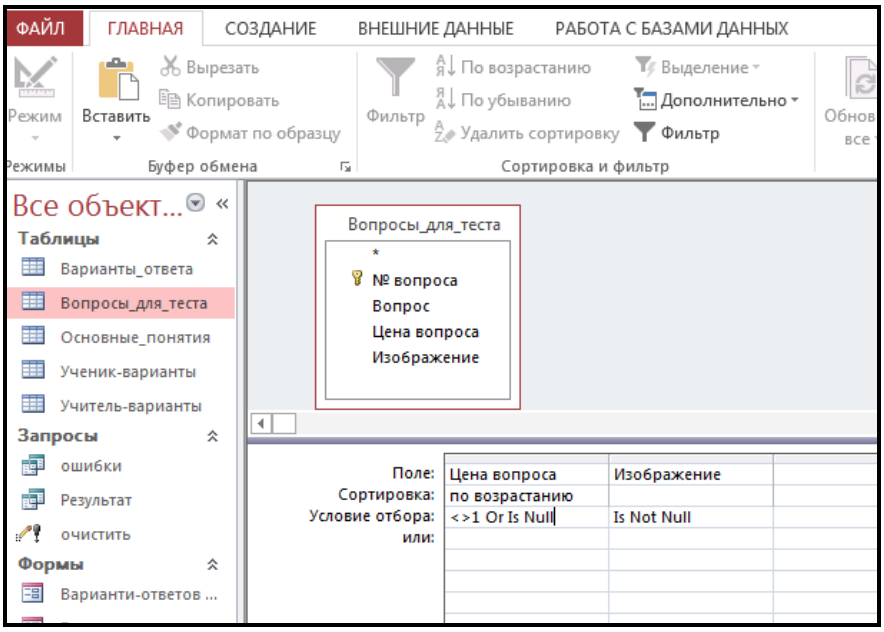


Рис. 7. Розширений фільтр

В рядку *Поле* указують поле по якому ставиться умова. Ім'я поля можна перетягнути з таблиці на стовпець бланка мишею або вказати за допомогою списку, що розкривається. У рядку *Сортировка* указують тип сортування, полем для сортування може бути будь-яке поле таблиці, у тому числі і те по якому немає умови.

✍ Якщо сортувань декілька, вони виконуються зліва-направо. Умови задані в одному рядку бланка виконуються водночас, тобто об'єднуються за допомогою операції «И», для об'єднання умов за допомогою операції «ИЛИ» умови записуються на різних рядках бланка.

Після запису умов дають команду застосувати фільтр – кнопка *Фильтр* на панелі інструментів. Операція *Очистить бланк* видаляє всі записані умови.

Розширений фільтр можна зберегти як окремий об'єкт, у вигляді простого Запиту, в цьому одна з переваг розширеного фільтру.

Лабораторна робота №2. Робота з базою даних у режимі таблиці

Мета: Придбання практичних навичок введення і редагування даних в таблицях, виконання операцій сортування, фільтрації, пошуку і заміни даних в таблицях. Вивчення правил запису виразів у фільтрах.

Завдання.

Запустіть Access, відкрийте створену раніше БД «Деканат». Виконайте наступні вправи.

1. Перегляд та додавання записів. Відкрийте таблицю «Студенти» у режимі таблиці.
   - 1.1. Сховайте всі стовпці крім: КодСт, НомСтБил, Прізвище, Ім'я, По батькові, НомГр.
   - 1.2. Змініть ширину стовпців так, щоб усі поля помістилися на екрані.
   - 1.3. Заповніть таблицю даними про 10 студентів з двох груп.

   ☞ *Вказівки.* Використовуйте поле зі списком для введення номера групи. Зверніть увагу на те, як програма контролює правильність уведених даних, як впливає на введення даних встановлений розмір поля.

   - 1.4. У таблиці «Групи» перегляньте інформацію про задану групу. До чого призводить зв'язування таблиць? Заповните в підлеглій таблиці кілька записів. Як змінюється при цьому порядковий номер записів?
   - 1.5. Відкрийте таблицю «Студенти», виконайте *Отобразить поля.*
   - 1.6. Перемістите в крайнє ліве положення поля «Прізвище» і «Ім'я», закріпите два цих поля.
   - 1.7. Доповніть записи інформацією у полях «Адреси», «Дата народження» тощо.

2. Сортування записів.

2.1. У таблиці «Студенти» відсортуйте записи по номеру групи в порядку зростання. Додайте кілька записів студентів однієї і тієї ж групи. Проаналізуйте результат.

2.2. Відсортуйте записи по двох стовпцях одночасно – спочатку по номеру групи, потім на прізвище в порядку зростання. Проаналізуйте результат.

3. Пошук записів.

3.1. У таблиці «Студенти» знайдіть записи студентів заданої групи

3.2. У таблиці «Студенти» знайдіть всі комірки, де зустрічається цифра 1

3.3. У таблиці «Студенти» знайдіть записи, прізвища яких починаються на «А» (або на іншу літеру).

4. Пошук та заміна**.**

4.1. У таблиці «Оцінки» всі «3» замінити на «5»

4.2. У таблиці «Студенти» номер групи «512», замінити на номер групи «522». Як впливає при цьому опція «Цілісність даних» на операцію заміни?

5. Фільтрація записів.

5.1. У таблиці «Студенти» задайте фільтр по виділеному так, щоб на екрані були записі заданої групи. Заповніть оцінками поля зв'язаної таблиці «Оцінки».

5.2. Використовуючи звичайні фільтри у таблиці «Студенти» виведіть списки

а) таких, що проживають у Сумах і мають телефон

б) студентів імена яких починаються на «А» і на «Б»

в) студентів імена яких починаються на «А» і прізвища на «Б»

г) студентів за віком менш 20 років

5.3. Використовуючи розширений фільтр у таблиці «Студенти» виведіть списки:

а) сімейних

б) не проживаючих у Сумах

в) народжених у 1997 році

г) студентів за віком менш 20 років й більш 17

? Контрольні запитання

1. Правила редагування даних у таблиці.
2. Правила керування розміщенням полів на екрані: закріплення, зміна ширини, переміщення, відображення стовпців.
3. Сортування даних у таблиці.
4. Пошук і заміна даних.
5. Фільтрація записів.
6. Способи встановлення фільтру.
7. Правила роботи фільтра по виділеному
8. Правила роботи фільтра по формі
9. Правила роботи розширеного фільтра
10. Правила запису умов фільтру для різних типів полів.

2.3. Запити.

Запити – найважливіший інструмент будь-якої СУБД, вони автоматизують операції пошуку, відбору даних, аналізу, оновлення базових таблиць тощо. Запити представляють вихідні дані для користування в межах установи. Призначення запитів:

1. вибірка записів, що задовольняють заданій умові
2. комбінування інформації, що міститься в різних таблицях з метою підбиття підсумків та аналізу
3. оновлення таблиць і включення в них нових записів.

Відповідно до функцій виділяють основні **типи запитів**:

1. *Запит на вибірку* витягує дані з однієї або декількох зв'язаних таблиць і відображає їх в результуючій таблиці. У складних запитах на вибірку присутні обчислювані віртуальні поля. Запити на вибірку є основним видом запитів, на їх основі побудована вся решта запитів.
2. *Групові або підсумкові запити* підсумовують дані з однієї або декількох таблиць у формі електронної таблиці, служать для аналізу даних і побудови діаграм і графіків, є незмінним, статичним набором даних.
3. *Перехресний запит*, це груповий запит, який відображає вибіркові дані в компактному і легкому для читання вигляді, у вигляді двовимірної матриці, так само як і зведені таблиці Excel.
4. *Запити на зміну* вносять в таблиці значні зміни, відмінити які не можна, у вікні БД такі запити помічені знаком оклику (!), вони підрозділяються в свою чергу на наступні підтипи:
   - 4.1. запит на видалення записів
   - 4.2. запит на додавання записів
   - 4.3. запит на створення таблиці
   - 4.4. запит на оновлення значень полів

Будь-який запит, що містить умови або вирази можна перетворити в запит з параметром. *Параметричні* запити – це запити, які при запуску пропонують ввести необхідний параметр або параметри у відповідних діалогових вікнах. Параметр може бути умовою для відбору записів, тоді користувач замість багатьох запитів на вибірку з різними умовами використовує один запит з параметром, який виводить різні дані залежно від параметру. Параметр може бути невідомою величиною для розрахунків, тоді результат розрахунків залежить від введеного параметру.

Виконаний запит це віртуальна таблиця, що відображає найсвіжіші дані витягнуті з базових таблиць відповідно умовам. Це тимчасовий об'єкт, який утворюється в пам'яті машини у момент виконання запиту. Запити оперують завжди свіжими даними, тому всі розрахунки здійснюються у запитах. Складні запити містять віртуальні поля, де йде обробка даних, обчислення за формулами. На диску у БД, якщо потрібно, зберігається інструкція запиту як окремий об'єкт.

Існують запити що оновлюються – динамічний набір даних і запити що не оновлюються – статичний набір даних. У запитах, що оновлюються,

можна редагувати дані і навіть додавати. Всі зміни зберігаються в базових таблицях, віртуальні поля буде перераховано. Ознака запиту, що оновлюється, символ ✱ (новий запис) в кінці таблиці. Як правило це запити на вибірку. Статичний набір даних призначений тільки для перегляду, редагувати дані не можна, це групові запити. Якщо змінити властивості запиту на вибірку можна перетворити його на статичний набір даних *(Свойства запроса / Тип набора записей / Статистический набор)*.

Записати умови запиту можна по-різному, в Access підтримуються два види запитів QBE і SQL.

Запит **QBE** – графічний бланк, який містить імена полів однієї або декількох зв'язаних таблиць у вигляді заголовків стовпців, в стовпцях можуть бути вирази; у рядках бланка указують ім'я таблиці, порядок сортування, вивід на екран, умови відбору. Бланк запиту нагадує бланк розширеного фільтру. На відміну від розширеного фільтру в запиті указують конкретні поля для виводу в результуючій таблиці. При заповненні бланка запит можна виконати, щоб переконатися в правильному відборі записів, для цього використовують кнопку на панелі інструментів *Выполнить.* Готовий запит зберігають вказавши ім'я, за замовчанням буде «Запрос1».

Запит **SQL** – послідовність інструкцій, написана на мові структурованих запитів, куди можуть входити вирази і статистичні функції SQL. Запит SQL має більш універсальне значення, запит до серверу, запит на об'єднання, можна створити тільки на мові SQL.

Один вид запитів швидко перетворюється в іншій, за допомогою кнопки на панелі інструментів запит можна проглянути у вигляді SQL інструкції, у вигляді бланка в режимі Конструктора і у вигляді результуючої таблиці в режимі виконання.

Запити можуть бути вкладеними, тобто один запит використовує дані іншого запиту. В цьому випадку спочатку виконується вкладений запит, а потім над цім набором даних виконується основний запит. Всі властивості вкладеного запиту впливають на виконання, наприклад, якщо вкладений запит був параметричним, параметри буде запрошено. Кількість вкладень не обмежена, але сильно уповільнює роботу.

Створення запитів.

Запити можна створити за допомогою Майстра або в Конструкторі (*Мастер запросов* або *Конструктор запросов* на стрічці *СОЗДАНИЕ*). Режим Конструктора – гнучкіший та універсальний.

Якщо вибрати ***Мастер запросов,*** у діалоговому вікні, що відкривається, потрібно вибрати один з варіантів:

1. Простий Запит (можливість створити запит на вибірку та запит груповий підсумковий)
2. Перехресний Запит
3. Записи, що повторюються
4. Записи без підлеглих

Переваги використання Майстра – простота, наочність. Недолік – неможливість вказати умову відбору. Створення запиту таким способом – це послідовне виконання всіх вказівок Майстра.

> ✍ Базові таблиці або запити, які є джерелом даних для Майстра запитів, мають бути зв’язані між собою. Інакше запит не виконається.

Часто разом з Майстром використовують Конструктор, щоб відредагувати запит створений за допомогою Майстра. Інколи, навпаки, треба у Конструкторі створити запит, на основі якого Майстер створює підсумковий запит.

Якщо вибрати ***Конструктор*** треба спочатку додати у вікно запиту ті таблиці або запити на підставі яких створюється запит, а потім заповнити графічний бланк запиту **QBE** (рис. 8).

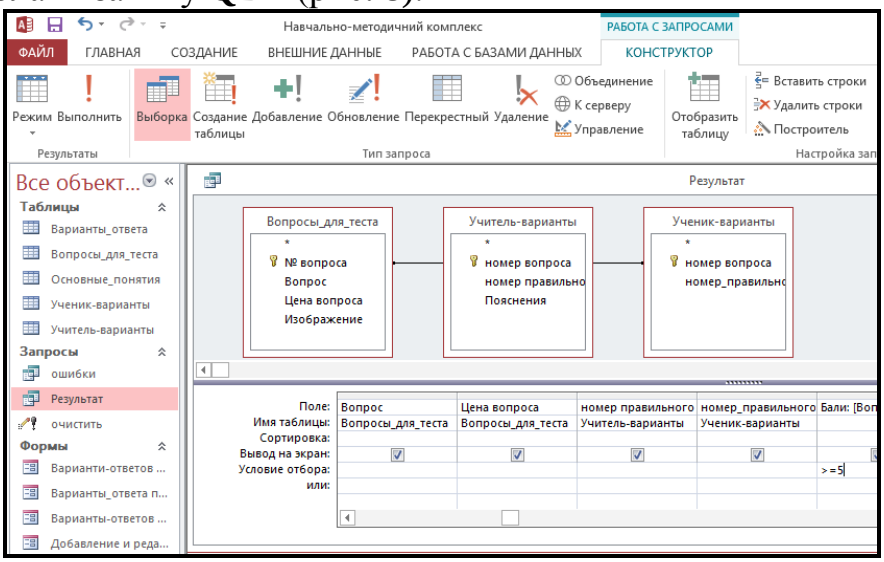


Рис. 8. Запит на вибірку в режимі Конструктора

> ✍ Таблиці або запити, які добавлено у вікно Конструктору запитів повинні бути зв'язані між собою. Інакше запит виконається неправильно! Якщо зв'язки вказані за допомогою Схеми даних, вони автоматично відображаються у всіх запитах. Якщо ні, зв'язки необхідно створити безпосередньо у вікні запиту, за тим же принципом, що і на схемі даних. При виконанні запиту на підставі таблиць з непрямими зв'язками, необхідно додати в запит весь ланцюжок зв'язаних між собою таблиць в потрібній послідовності.

Для заповнення бланка ***Запиту на вибірку*** необхідно:

1. Вибрати поля для Запиту, це можна зробити декількома способами:
   - Вибрати із списку, що розкривається, ім'я таблиці а потім ім'я поля у відповідних рядках бланка Запиту
   - Двічі клацнути в табличці на імені поля
   - Перетягнути на стовпець бланка ім'я поля, або декілька штук, або символ * – що означає всі поля, при цьому ім'я таблиці автоматично з'являється в бланку.

2. Визначити вивід на екран полів Запиту у відповідному рядку бланка. Іноді поле бере участь в Запиті, але не виводиться на екран, наприклад служить для завдання умови.
3. Вказати сортування для необхідних полів, вибравши із списку, що розкривається, один з варіантів: за збільшенням, за зменшенням, відсутній. За замовчанням сортування йде по ключових полях, для декількох полів зліва направо.
4. Замовити умови відбору у відповідному рядку або декількох рядках.

✍ Якщо умови вказані в одному рядку – вони об'єднані логічною операцією «И», якщо в різних рядках – логічною операцією «АБО».

Якщо у виразі умови присутнє значення поля воно має бути в квадратних дужках [ ].

5. Переглянути результат, виконати Запит, використовуючи кнопку *Выполнить*
6. Відредагувати Запит повернувшись в режим роботи з бланком в Конструкторі або в режимі SQL. Можна змінити: послідовність стовпців, виділяючи і перетягуючи їх; змінити ширину стовпців, перетягуючи за заголовок. Змінити властивості поля Запиту (підпис-заголовок, формат, маску) можна через контекстне меню стовпця. Змінити назви-підписи полів не вдасться, якщо у поля вже є підпис – властивість, яка вказана в базовій таблиці.
7. Зберегти Запит як об'єкт, давши йому ім'я, за замовчанням –Запрос1.

Відредагувати створені раніше Запити можна в режимі Конструктора або режимі SQL в будь який момент.

Щоб створити ***обчислюване віртуальне поле*** в стовпець вводять нове ім'я поля та двокрапку (за замовчанням Выражение1:), далі вказують вираз для підрахунку, вводячи його безпосередньо або використовуючи «Будівника виразів». Наприклад для обчислення середнього балу – СрБал:(Ос1+Ос2+Ос3)/3. Поля, які вказано у виразі повинні бути серед джерела даних Запиту, тобто бути присутніми в доданих таблицях або запитах.

Щоб перетворити запит на ***Запит з параметром*** треба вказати які параметри треба ввести при запуску й відповідно їх використати. При запуску Запиту з параметром на екран виводиться повідомлення про введення параметру. Це повідомлення в інструкції запиту задається у квадратних дужках […], та може бути або у виразі умови, або у виразі для розрахунків. Значення введеного параметру підставляється у вираз, запит виконується з цім значенням.

Найчастіше параметр використовують в умовах Запиту. Наприклад, замість декількох запитів для різних груп студентів, доцільніше створити один з параметром, де параметром є номер групи. Такий Запит можна виконувати при різних значеннях параметра, який задає умову відбору записів. В інструкції запиту в рядку «Умови відбору» замість конкретної

умови вводять в [ ] повідомлення про введення параметра (не більше ніж 50 символів, зайві відсікаються).

✐ Приклади виразів-умов для запитів з параметрами:

[введіть номер групи]

between [початкова дата] And [кінцева дата]

Like [введіть номер групи або <Enter>для всіх груп] &” *”

Якщо параметрів декілька, тоді порядок введення параметрів за замовчанням зліва направо відповідно до бланка запиту. Змінити черговість параметрів та інші їх властивості можна викликавши *Параметры* на стрічці *РАБОТА З ЗАПРОСАМИ / КОНСТРУКТОР*. Запит з невідомими параметрами не виконається, тобто записі не буде знайдено.

✍ Повідомлення в [ ] повинно відрізнятися від імені поля, інакше це не є невідомим параметром, який треба вводити. Будь яке слово, що не співпадає з іменем поля в [ ] у виразі сприймається як невідомий параметр, який треба запитувати. Цим користуються коли треба здійснити розрахунки для змінних величин.

Щоб змінити ***властивості Запиту*** в режимі Конструктора викликати контекстне меню запиту – клацнути правою кнопкою миші на сірому полі вище за бланк Запиту, або виконати *Страница свойств/Общие*. Вказати необхідні властивості.

✐ Властивості запиту необхідно змінювати, якщо наприклад, виводиться не всі записи, а тільки 10. Це властивість *Набор значений*, або кнопка *Вернуть* на стрічці *РАБОТА З ЗАПРОСАМИ / КОНСТРУКТОР*. Якщо відсортувати записи у Запиті по середньому балу за зменшенням, та вказати властивість «*Набір значень 10*», отримаємо список 10 кращих за успішністю студентів.

✐ Властивість «*Унікальні значення: Так»* замовляє відображення унікальних значень, виключає ті, що повторюються, а також перетворює Запит в статичний набір даних, такий запит використовується для перегляду і аналізу.

Лабораторна робота №3. Створення запитів на вибірку

Мета: придбання практичних навиків створення запитів на вибірку, параметричних запитів та запитів з обчислюваними полями.

Завдання.

Запустіть Access, відкрийте створену раніше БД «Деканат». Виконайте наступні вправи. Кожному створеному запиту дайте ім'я за змістом: Відмінники, або Список з оцінками, або Сумчани.

1. Створення запиту за допомогою Майстра

1.1. Створіть запит для виводу прізвищ, імен і повних адрес студентів, використовуючи як джерело даних таблицю «Студенти».

1.2. Створіть запит для виводу прізвищ, імен і оцінок студентів по всім предметам із заліками, використовуючи таблиці «Студенти» і «Оцінки».

2. Створення запитів за допомогою Конструктора.

2.1. Створіть запит для виводу списку відмінників, вивести прізвище, ім'я, номер групи, оцінки.

2.2. Створіть запит для виводу даних про студентів, що мають двійки на іспитах, хоча б один із предметів.

2.3. Створіть запит для виводу списку тих, хто проживає в Сумах, вивести прізвище, ім'я, адресу, телефон.

2.4. Модифікуйте таблицю «Студенти» так, щоб додати дані про знання іноземної мови і про володіння комп'ютерною технікою. Створіть запит для виводу списку студентів, що володіють знаннями комп'ютерної техніки та іноземною мовою.

3. Створення запитів з обчислюваними полями

3.1. Створіть запит для виводу даних про успішність студентів заданої групи, у запиті виведіть групу, номер студ. білета, прізвище, ім'я, оцінки і середній бал студента. Поле середній бал обчислюється. Змініть формат поля середній бал для виводу з точністю два знака після коми.

3.2. Скопіюйте створений запит та перетворіть у запит з параметром, де запитується номер групи.

3.3. Створіть запит для виводу списку кращих 10 студентів за успішністю.

? Контрольні запитання

1. Призначення запитів.
2. Типи запитів.
3. Що являє собою запит?
4. У чому відмінності запиту від фільтра?
5. Алгоритм створення запиту за допомогою Конструктора.
6. Алгоритм створення запиту за допомогою майстра простих запитів.
7. Яку роль грають зв'язки між таблицями при побудові запитів?
8. Як змінити порядок виводу знайдених записів?
9. Як змінити кількість виводу знайдених записів?
10. Параметричні запити, їх особливості.

2.4. Створення групових запитів та запитів на зміну

Підсумкові та перехресні запити відносять до групових запитів, їх використовують для підведення підсумків та аналізу.

Групові запити групують (об'єднують в групи) записи по однакових значеннях поля, або декількох полів. Якщо в полі немає значень, що повторюються, угрупування по ньому неможливе. Групові запити – це статичний набір даних. Створювати групові запити можна за допомогою Майстра й за допомогою Конструктора.

Створення підсумкових запитів за допомогою Майстра

Майстер простих запитів дозволяє створити як запит на вибірку так і підсумковий запит, вибір варіантів на 2-му кроці Майстра.

1 крок: Вибрати поля для запиту. Допускається вибір полів з декількох таблиць або інших запитів зв'язаних між собою.

2 крок: Вибрати варіант виведення кожного поля кожного запису для запиту на вибірку, або *Итоговый* для підсумкового запиту. У останньому випадку кнопка з командою «*Итоги*» – активна. Клацнувши по ній указують підсумкові функції для числових полів, і підрахунок значень для будь-яких типів полів.

3 крок: Дати ім'я Запиту, за замовчанням це ім'я таблиці, на основі якої створений запит. Проглянути запит – виконати, або відредагувати його в режимі Конструктора. При закриванні запиту – зберегти його.

Недоліком створення підсумкового запиту за допомогою Майстра є неможливість вказати конкретне поле групування записів та неможливість провести підсумковий аналіз для полів які не є числовими. Всі поля які не є числовими автоматично стають полями групування записів, в тому числі й ті, які не мають значень, що повторюються. Це призводить к відсутності групування взагалі й неправильному аналізу.

✍ При створенні підсумкового запиту за допомогою Майстра простих запитів уважно виконуйте вибір полів для запиту: це мають бути поля числового типу й лише одне поле групування зі значеннями, що повторюються, яке може бути не числовим.

Створення підсумкових запитів за допомогою Конструктора

Спочатку створюється запит на вибірку, потім перетворюється на груповий – кнопка *Итоги* на панелі інструментів. У бланку запиту з'являється рядок *Групповая операция* для кожного поля стовпця в цьому рядку можна вибрати один з варіантів:

1. *Группировка* – по полю групуються записи для подальших підсумкових обчислень
2. Підсумкові статистичні функції (9 штук) – здійснюють математичні операції або операції вибору над значеннями в полі, а саме:

Avg середнє значення від числових значень в полі
Sum сума числових значень в полі
Max максимальне з числових значень в полі
Min мінімальне з числових значень в полі
Var дисперсія розподілу числових значень в полі
StDev стандартне відхилення числових значень в полі
Count підрахунок кількості не порожніх значень в поле, будь-який тип поля
First перший зі списку, будь-який тип поля
Last останній зі списку, будь-який тип поля

При виборі функції в підписі поля стовпця з'явиться назва функції і двокрапка, наприклад: Count:([Прізвище]), або Sum:([Стипендія]), що відображає операцію з даними в полі; а результат з'явиться у комірці при виконанні запиту.

3. *Выражение* – указується якщо необхідно обчислити складнішу підсумкову функцію з декількома операторами і статистичними функціями, використовується для обчислюваних полів. Необхідно привласнити ім'я полю (за замовчанням *Выражение*:) і вказати необхідні операції.
4. *Условие* – указується для поля-критерію групування. Якщо є умови, що визначають вибірку записів ***перед*** угрупуванням, умова вводиться в рядку *Условие отбора.* При цьому *Вывод на экран* значень відміняється.

Перевірка умов і сортування здійснюватимуться також після групування значень і підсумкових обчислень. У рядках бланка *Сортировка*, *Условие отбора, или* указують необхідні вирази. Запит виконують, редагують, зберігають.

Перехресний Запит представляє підсумкові дані у вигляді двовимірної матриці, групування здійснюється по полю-заголовки стовпців, і по полю-заголовки рядків, підсумкові дані відображаються в комірках таблиці. Простіше створювати перехресний запит за допомогою Майстра (*СОЗДАНИЕ* / *Мастер запросов* / Перекрестный запрос)

Створення перехресних запитів за допомогою Майстра

1 крок: виберіть одну таблицю або один запит, поля якого братимуть участь в перехресному запиті.

✍ При створенні перехресних запитів за допомогою Майстра для включення полів з декількох зв’язаних таблиць або запитів, а також якщо є умови відбору записів перед групуванням, створіть заздалегідь запит на вибірку.

2 крок: виберіть поля, значення яких буде використано як заголовки рядків, не більше трьох полів, враховуйте сортування, якщо полів декілька.

3 крок: виберіть одне поле, значення якого буде використано як заголовки стовпців.

4 крок: виберіть одне поле для підрахунку підсумків і вкажіть тип обчислень. Залежно від типу поля будуть запропоновані різні статистичні функції. Підключіть параметр – обчислити підсумкове значення для кожного поля рядка.

5 крок: вкажіть ім'я Запиту, виведіть його для перегляду або зміни структури в режимі Конструктора.

При закриванні Запиту збережіть його.

Створення перехресного запиту за допомогою Конструктора

1. Створюють звичайний запит на вибірку на основі однієї або декількох таблиць, указують умови відбору що виконуються ДО групування записів.
2. Перетворюють запит в перехресний – команда *Тип запроса / Перекрестный*. У бланку Запиту замість рядка *Вывод на экран* з'являються рядки *Групповая операция* та *Перекрестная таблиця.*

В цих рядках необхідно вказати:

2.1. для відповідного поля – *Заголовки строк*, (можна для декількох полів, не більше трьох, в потрібному порядку), в рядку групова операція вказати *Группировка;*

2.2. для відповідного поля – *Заголовки столбцов*, лише одно, в рядку групова операція вказати *Группировка*.

2.3. для поля, по якому здійснюється вибір значень і підбиття підсумків, указуємо *Значения*, в рядку групова операція указуємо статистичну функцію або вираз.

3. У рядку *Условие отбора* указують умови для виведення даних в перехресному запиті після групування

3.1. умова в полі заголовки рядків виключає невідповідні значення заголовків рядків

3.2. умова в полі заголовки стовпців виключає невідповідні значення заголовків стовпців

3.3. умови в додаткових стовпцях-критеріях групування указують умови що виконуються ДО угрупування. При цьому в рядку перехресна таблиця – порожньо, в рядку групова операція – умова. Ці параметри з'являються автоматично, якщо умова відбору записів для угрупування була вказана в запиті на вибірку.

4. У рядку *Сортировка* указують порядок виведення заголовків рядків і стовпців на екран.

Після заповнення бланка запит виконують, редагують, зберігають.

✍ ***Запити на зміну*** вносять зміни в базові таблиці які не можна відмінити! На вкладці *Все объекты Access* вони відмічені піктограмою із знаком оклику, будьте уважні при виконанні таких запитів. Проглянути зміни можна відкривши базові таблиці.

Використовуються запити на зміну рідше, ніж запити на вибірку.

Всі запити на зміну створюються в два етапи: спочатку створюється і тестується запит на вибірку, а після перетворюється в запит на зміну.

1. ***Запит на видалення*** створюють при видаленні записів з декількох зв'язаних таблиць, а також при необхідності часто виконувати видалення записів з таблиці по деякій умові.

Послідовність дій:

1.1. створити запит на вибірку, в який включити **всі** поля таблиці, шляхом перетягування в стовпець бланка запиту символу *, що означає – всі поля;

1.2. додати поля, по яких указують умови, у наступні стовпці бланка, вказати умови;

1.3. виконати запит, перевірити правильність відібраних для видалення записів;

1.4. перетворити запит в запит на видалення, давши команду меню *Тип запроса / Удаление.* У бланку запиту рядок *Сортировка* заміниться на *Удаление;*

1.5. підтвердити при виконанні запиту видалення певної кількості записів;

1.6. зберегти запит якщо є необхідність.

✍ Записи видаляються цілком. Для видалення записів з декількох зв'язаних таблиць треба підключити параметр «Каскадне видалення» в діалоговому вікні властивостей зв'язку на схемі даних.

2. ***Запит на оновлення*** створюють для автоматичного оновлення полів великої кількості записів, коли іншими засобами, наприклад – «Пошук і заміна», зробити це неможливо. Запит на оновлення за допомогою виразів вбудованих в бланк запиту вносить зміни в записі.

✎ Наприклад, вираз [Стипендія]+50, застосовують для підвищення стипендії на 50 грн., де [Стипендія] – ім'я поля.

Запити на оновлення звичайно не зберігають, зберігають тільки якщо доводиться одне і теж оновлення проводити багато разів. Для зв'язаних таблиць підключають параметр «Каскадне оновлення» в діалоговому вікні властивостей зв'язку на схемі даних.

Послідовність дій:

2.1. створити запит на вибірку, в який входять поля для запису умов і ті, що оновлюються, задаються умови, запит тестується;

2.2. перетворити в запит на оновлення, команда *Тип запроса / Обновление*. У бланку запиту замість рядків *Сортировка* і *Вывод на экран* з'являється рядок *Обновление;*

2.3. вказати в рядку *Обновление* для поля, що оновлюється, нове значення, або вираз для нового значення;

2.4. виконати запит, підтвердити оновлення записів.

2.5. зберегти запит при необхідності.

3. ***Запит на створення нової таблиці*** створює нову таблицю із структурою початкового запиту і тими записами, які були відібрані. Послідовність дій:

3.1. створити і протестувати запит на вибірку;

3.2. перетворити в запит на створення таблиці *Тип запроса / Создание таблицы...*

3.3. вказати ім’я нової таблиці;

3.4. підтвердити створення нової таблиці з вказаним числом записів;

3.5. відредагувати нову таблицю в режимі Конструктора, визначити ключове поле;

3.6. організувати зв'язок таблиці з іншими таблицями у вікні схеми даних.

4. ***Запит на додавання записів*** використовують у разі неможливості скопіювати записи з однієї таблиці в іншу через буфер обміну, що вимагає строгої відповідності структури двох таблиць. Запит на додавання записів з однієї таблиці в іншу може використовувати складні умови для вибірки записів, може використовувати меншу кількість полів, чим міститься в таблиці, в яку додають записи. Типи полів запиту і таблиці повинні співпадати. Послідовність дій:

4.1. створіти запит на вибірку на основі першої таблиці, протестувати;

4.2. перетворити в запит на додавання *Тип запроса / Добавление…*
4.3. вказати ім'я таблиці, в яку додають записи, в бланку запиту з'явиться рядок *Добавление;*
4.4. вказати імена полів таблиці які відповідають полям запиту у рядку *Добавление*;
4.5. підтвердити додавання записів
4.6. зберегти при необхідності

Лабораторна робота №4. Створення групових запитів та запитів на зміну

Мета: придбання практичних навиків створення підсумкових і перехресних запитів для аналізу даних таблиць, створення запитів оновлення таблиць.

Завдання.

Запустіть Access, відкрийте створену раніше БД «Деканат». Виконайте наступні вправи. Кожному створеному запиту дайте ім'я за змістом.

1. Створення групових запитів

1.1. Створіть запит для підрахунку кількості студентів, що володіють іноземною мовою.

1.2. На основі Запиту про успішність студентів створіть запит, що виводить для різних груп у стовпцях:

а) загальну кількість студентів;
б) успішність по окремих предметах;
в) успішність у цілому;
г) мінімальне значення успішності;
д) максимальне значення успішності.

1.3. Створіть перехресний запит для виводу даних про успішність окремих груп

2. Створення запитів на відновлення.

Модифікуйте таблицю «Студенти», додавши поле, «Стипендія» (тип грошовий).

2.1. Створіть запити для призначення стипендії всім студентам за наступними правилами:

100 грн. – тим, у кого середній бал дорівнює 5,0.
50 грн. – тим, у кого середній бал більше дорівнює 4,5.
30 грн. – тим, у кого середній бал менше 4,5.

2.2. Створіть запит для підвищення стипендії на 50 грн. студентам, що не проживають у Сумах.

3. Створення запитів на видалення.

3.1. Створіть запит на видалення записів, відрахованих за неуспішність студентів (круглих двієчників).

3.2. Створіть запит на видалення записів студентів заданої групи.

4. Створення параметричних запитів.

4.1. Створіть запит для виводу списку студентів визначеного року народження. Перетворіть у запит з параметром.

? Контрольні запитання:

1. Групові запити, їх особливості
2. Запити на зміну даних у таблицях, їх особливості.
3. Алгоритм створення запиту на видалення записів
4. Алгоритм створення запиту на оновлення записів
5. Алгоритм створення запиту на додавання записів
6. Алгоритм створення запиту на створення таблиці
7. Алгоритм створення підсумкового запиту за допомогою майстра
8. Алгоритм створення перехресного запиту за допомогою майстра
9. Алгоритм створення підсумкового запиту у Конструкторі
10. Алгоритм створення перехресного запиту у Конструкторі

2.5. Форми

Форми служать для організації користувальницького інтерфейсу додатку БД. За допомогою форм зручно додавати дані в таблицю, в декілька взаємозв'язаних таблиць, а також переглядати і редагувати дані. У режимі перегляду для форми можна використовувати операції Пошук, Сортування, Фільтрація. Правильно сконструйована форма прискорює введення даних, зводить до мінімуму помилки введення оператора. Форми створюють для базових таблиць, для запитів, іноді форми виводять на друк. Форми з елементами керування використовують для діалогу з користувачем, для керування роботою додатку БД.

У вікні форми для перегляду даних містяться два **типи елементів**:

1. елементи зв'язані, приєднані до полів джерела даних, наприклад, до полів таблиці або запиту: це поля для введення і редагування даних, списки, що розкриваються, поля із списком, групи перемикачів пов'язаних з деяким полем і т.д.
2. елементи вільні, не залежні від джерела даних, елементи керування та оформлення форми: це командна кнопка, набір вкладок, прямокутна рамка для об'єднання елементів в логічну групу, лінія для розділення на логічні зони, напис, малюнок і т.д.

Головний елемент форми – **поле**, має чотири основні типи:

1. однорядкові поля, зокрема поля із списком, приєднані до полів таблиці або запиту – займають у формі один рядок;
2. багаторядкові поля, приєднані до полів типу МЕМО або довгий текст, виводять інформацію в декілька рядків;
3. обчислювані поля містять вирази, починаються із знаку =, можуть бути вільними або зв'язаними, їх використовують для обробки інформації, значення в обчислюваних полях редагувати неможливо;
4. вільні поля для введення дат, використовуються в процедурах VBA.

Правильно сконструйована форма окрім області даних має *Заголовок, Примечание*. Якщо елементів форми багато, їх розподіляють по вкладках.

Розташування елементів в області даних може бути чотирьох видів:

1. *форма у стовпець* – поля розташовані в один стовпець, іноді у дві колонки. Кожен запис на окремій сторінці форми, переміщення по записах здійснюється за допомогою рядка навігації;
2. *вирівняна форма* – поля розташовані в один стовпець або щільно в декілька стовпців. Кожен запис на окремій сторінці форми, переміщення по записах здійснюється за допомогою рядка навігації;
3. *таблична форма* – відображає дані аналогічно базовій таблиці. У вікні форми багато записів, використовується для підлеглих форм;
4. *стрічкова форма* – гнучкіший варіант табличної. У вікні форми декілька записів, поля введення можна розміщувати в потрібному порядку, редагувати в режимі Конструктора колір, шрифт, розмір полів введення і т.д.

Для окремої таблиці, крім вказаних вище варіантів розташування елементів у вікні форми, часто використовують комбінований варіант – *розподілена форма*. Одна частина вікна є формою у стовпець, друга частина вікна – базова таблиця, рядок навігації спрацьовує і для форми, і для таблиці. Розподілена форма зручна тим, що об’єднує переваги обох способів роботи з даними. Орієнтація розподіленої форми може бути різною: таблиця зверху, знизу, зліва або справа. Ці параметри вказують у властивостях форми у режимі Конструктора.

Для таблиць зв'язаних відношенням один-до-одного створюють одиночну форму, в якій присутні поля обох таблиць. Для таблиць зв'язаних відношенням один-до-багатьох можна побудувати три види представлення даних:

- *одиночна,* поля обох таблиць присутні на кожній сторінці форми, поля з боку таблиці «один» повторюються для всіх зв'язаних записів таблиці з боку «багато».
- *підлегла,* створюються головна і підлегла форми, головна містить поля для таблиці з боку «один», підлегла містить зв'язані записи з боку таблиці «багато». У вікні головної форми можна проглянути зв'язані записи з підлеглої таблиці.
- *зв'язана,* аналогічна попередньої, тільки підлегла форма відображається у вікні головної у вигляді командної кнопки, відкриває записи при клацанні на цій кнопці. Економить місце на екрані.

У підлеглу форму теж може бути вкладена форма, всього можливо два вкладення.

Створення форм різних видів – процес, який найбільше автоматизований у БД Access. Створити форму – означає створити та зберегти у БД макет форми у вигляді окремого об’єкту. При запуску форми у вікні області даних виводяться данні з базових таблиць. Основних способів створення форми три: автоматичний (Автоформа), Майстер (покрокове виконання вказівок), Конструктор (з нуля). Для редагування макету форми включають режим *Конструктор*, або *Режим макета*.

На стрічці *СОЗДАНИЕ/ФОРМЫ* присутні команди, які використовують для створення різних видів форм:

Форма

Конструктор форм

Пустая форма

Мастер форм

Навигация

Другие формы…(Несколько элементов, Таблица, Разделенная форма, Модальное диалоговое окно)

Форма. Автоматично створюється найбільш оптимальний (на думку програмістів) тип форми для вибраної таблиці або запита. У формі присутні всі поля таблиці в тому ж порядку що й в таблиці. Стиль оформлення класичний. Для таблиці, яка зв'язана з підлеглою обирається тип підлеглої форми. Цей спосіб зберігає час, але не завжди задовольняє користувача, доводиться редагувати форму у режимі Конструктора.

Мастер форм. Створення форми за допомогою Майстра – це покрокове виконання вказівок Майстра з можливістю відмінити задані параметри.

1. Перший крок Майстра – вибір полів для форми, дозволяється вибір з декількох таблиць або запитів зв'язаних між собою.
2. Другий крок Майстра – вибір типу представлення даних для таблиць зв'язаних відношенням один-до-багатьох (підлегла або зв'язана форма), цей крок присутній лише для цього типу зв'язку.
3. Третій крок Майстра – вибір типу розміщення даних: в стовпець, стрічковий або табличний.
4. Четвертий крок Майстра – введення імені або імен форми, головній та підлеглої. Вибір варіанту для подальшої роботи з формою:

- відкрити форму для перегляду або вводу даних
- відкрити для редагування макету у режимі Конструктора
- Готово (варіант для закриття зі збереженням форми в БД).

Якщо деякі значення полів не видно, Майстер автоматично створює смуги прокрутки, горизонтальну та вертикальну, також автоматично створюється рядок навігації щодо переміщення по записах.

Конструктор форм. Створення форми у режимі Конструктора трудомісткий довготривалий але найбільш гнучкий процес створення макету. Частіше у режимі Конструктора редагують форми які створені автоматично або за допомогою Майстра. При цьому використовують команді на стрічці *ИНСТРУМЕНТЫ КОНСТРУКТОРА ФОРМ / КОНСТРУКТОР, УПОРЯДОЧИТЬ, ФОРМАТ* а також команди контекстного меню об'єктів. В необхідних випадках підключають спеціальній майстер для створення поля зі списком, командної кнопки, групи перемикачів та ін. Кнопка *Использовать мастера* на панелі *Элементы управления* має бути підключена. Створення елементів здійснюють перетаскуванням елемента в область даних. Положення елементів змінюється шляхом перетаскування мишею. Властивості елементів змінюють через контекстне меню *Свойства.* Контекстне меню області даних

дозволяє змініть фон. Для роботи с *Заголовком* та *Примечанием* команда *Колонтитулы/Заголовок*. Розмір елементів та області даних, областей заголовка та примітки установлюється або через контекстне меню *Макет*, або просто перетягуванням мишею. Сітка допомагає вирівняти елементи. У режимі перегляду перевіряєте роботу елементів форми. Приклад форми наведено нижче на рис. 9-10.

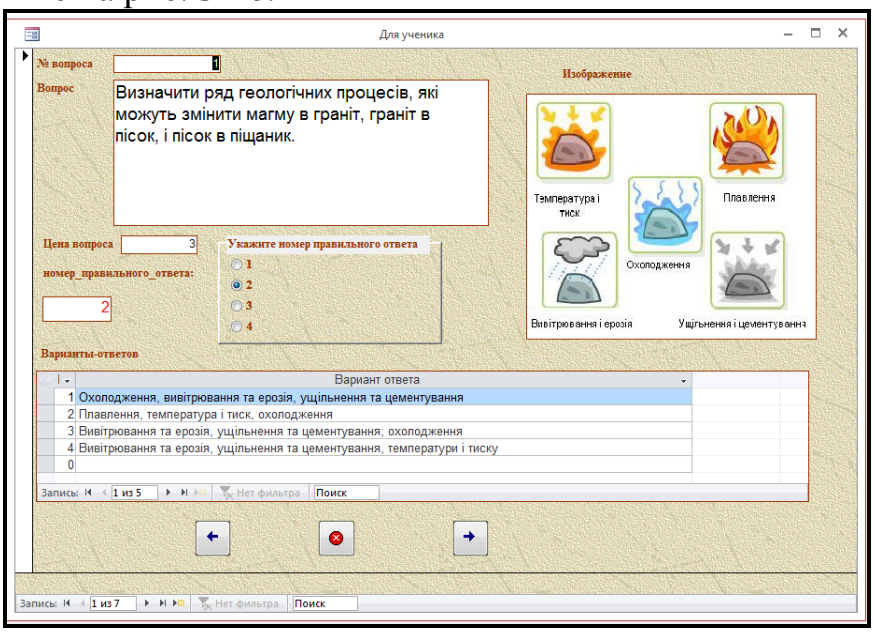


Рис. 9. Форма «Тестирование» у режимі перегляду

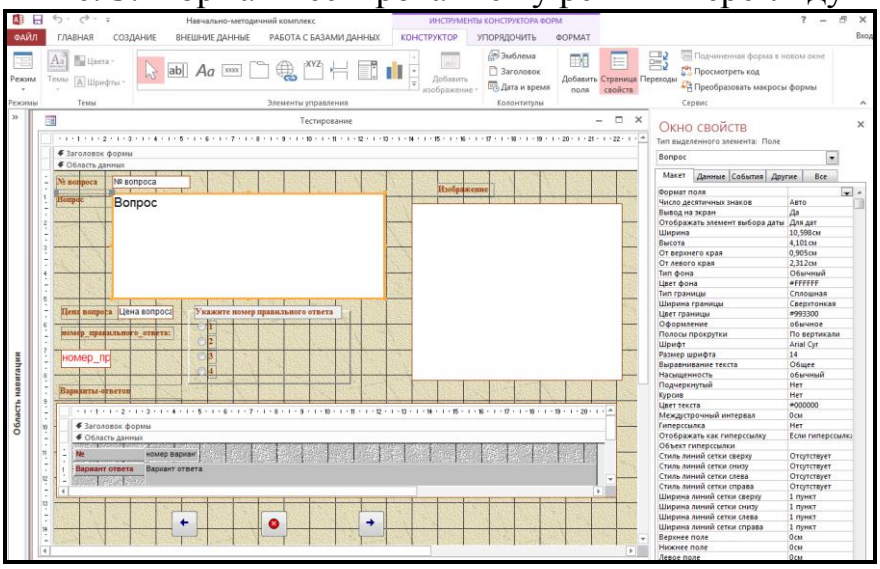


Рис. 10. Форма «Тестирование» у режимі Конструктора

Лабораторна робота №5. Створення форм

Мета: Придбання практичних навичок створення форм, використання форм для введення і редагування записів, для пошуку і фільтрації даних.

Завдання.

Запустіть Access, відкрийте створену раніше БД «Деканат». Виконайте наступні вправи. Кожній створеній формі дайте ім'я за змістом.

1. *СОЗДАНИЕ/Форма* .

1.1. Створити автоматичну форму для роботи із записами таблиці «Студенти», зберегти за назвою «Студенти».

1.2. Створіть форму для одночасного заповнення таблиць «Студенти» і «Оцінки»

1.3. Створіть форму для одночасного заповнення таблиць «Групи» і «Студенти»

2. *СОЗДАНИЕ/Мастер форм*.

2.1. За допомогою Майстра створити форму для роботи із записами таблиці «Оцінки»

☞ *Вказівки.* Параметрі форми: містить усі поля, поля розташовані в один стовпець, назва форми «Оцінки».

2.2. Створіть форму для одночасного заповнення таблиць «Студенти» і «Оцінки» за допомогою Майстра.

2.3. Створіть головну і підлеглу форму для таблиць «Групи» і «Студенти» за допомогою Майстра.

2.4. Створіть зв'язані форми для таблиць «Групи» і «Студенти» за допомогою Майстра.

3. *СОЗДАНИЕ/Конструктор форм*

3.1. Виконайте редагування в Конструкторі форми для таблиці «Оцінки», яка створена за допомогою Майстра. Додайте групу перемикачів для заповнення оцінок на Екзам1.

3.2. Створити форму для роботи із записами таблиці «Група».

☞ *Вказівки.* Ввести Заголовок форми та Примітку, вставити відповідний змісту малюнок, створити елемент «поле зі списком» для поля спеціальність, для обрамлення групи полів використати елемент «Прямокутник» з «Панелі елементів».

4. *СОЗДАНИЕ/ Другие формы/ Разделенная форма*

4.1. Створити розподілену форму для роботи із записами таблиці «Студенти».

4.2. Створити розподілену форму для роботи із записами таблиці «Групи».

5. Використання форм

5.1. Протестуйте створені форми переміщаючись по записах, вводячи і редагуючи значення полів, додайте записи, видалите деякі з них.

5.2. Попрацюйте з формою «Для таблиці Студенти» використовуючи пошук, сортування і фільтрацію записів.

? Контрольні запитання

1. Призначення форм.
2. Елементи форм та їх призначення.
3. Види форм, їх особливості.
4. Режими роботи з формами.
5. Способи створення форм.
6. Створення форми за допомогою Майстра.

7. Створення форми за допомогою Конструктора.
8. Призначення і використання кнопок панелі елементів у Конструкторі форм.
9. Операції переміщення, зміни розмірів, видалення елементів керування форми.
10. Редагування записів за допомогою форм, пошук і фільтрація.

2.6. Звіти

Звіти в Access використовуються для виводу даних на екран або на принтер, вони представляють вихідні дані для користування за межами установи. Звіт може представляти дані з однієї таблиці, з декількох зв'язаних таблиць, або з деякого запиту. Операції які можливі над даними в звітах: групування, сортування, підведення підсумків та статистичного аналізу, представлення даних у вигляді діаграм та графіків.

Існує два основні ***типи звітів***:

1. Стрічковий (*Ленточный*) найбільш поширений
2. Для розсилки (шаблони листів та поштові наклейки)

Створити звіт – означає створити та зберегти у БД макет звіту у вигляді окремого об'єкту. При відкриванні звіту виконується збір даних, формування звіту, автоматично додаються номера сторінок та указується системна дата на кожної сторінці. Таким чином, виконаній звіт – це документ цілком готовий до друку, який містить необхідні дані та параметри форматування печатної сторінки, у ньому виводяться данні з базових таблиць та підсумкові дані. Перегляд та друк звіту можливий тільки при встановленому принтері.

Основних способів створення звіту три: автоматичний (Автозвіт), Майстер (покрокове виконання вказівок), Конструктор (з нуля). На стрічці *СОЗДАНИЕ/ОТЧЕТЫ* присутні команди, які використовують для створення різних видів звітів:

Отчет
Конструктор отчетов
Пустой отчет
Мастер отчетов
Наклейки

Отчет. Автоматично створює звіт для вибраної таблиці або запита. У звіті присутні всі поля таблиці в тому ж порядку, стиль оформлення класичний. Цей спосіб зберігає час, але не завжди задовольняє користувача.

Мастер отчетов використовують для збору даних, групування даних та статистичного аналізу, визначення макету, формату, параметрів сторінки.

Конструктор отчетов використовують для створення або редагування макету звіту.

Наклейки – це спеціальний майстер для створення звітів у стовбець (або декілька стовбців), які використовують як листи для розсилки або наклейки.

Для редагування макету звіту включають режим *Конструктор*, або *Режим макета*. Режим *Предварительный просмотр* відображає сторінки, які

будуть роздруковані на принтері. Режим *Представление отчета* зручний для відображення звіту в електронному вигляді.

Створення звіту за допомогою Майстра.

1. Перший крок Майстра: вибір полів з таблиць або запитів, допускається вибір декількох зв'язаних таблиць або запитів.
2. Другий крок Майстра: вибір рівня групування записів, що задає структуру звіту та дозволяє вказати статистичні функції.

> ✍ Поле для групування повинно містити значення які можна групувати (наприклад дати – по місяцям, по тижням, по рокам; числа – або повторюються, або групуються по інтервалам; текст – або повторюється, або групується по інтервалам). Для того щоб вибрати інтервали групування для деякого поля вказують команду *Группировка*. Якщо вибрати декілька полів можна міняти рівні групування. Якщо у звіті немає числових полів, або не вибрано рівень групування, підведення підсумків неможливо на наступному кроці.

3. Третій крок Майстра: вибір порядку сортування записів, дозволяє вибрати до чотирьох полів для сортування та тип сортування для кожного поля.

> ✍ Для підведення підсумків скористуйтесь командою *Итоги*. Вкажіть поля для аналізу та конкретні статистичні функції: Sum, Avg, Min, Max. Уточніть вивід даних: *данные и итоги* або *только итоги*. Порахуйте відсоток, якщо потрібно.

4. Четвертий крок Майстра: вибір типу макета та орієнтації сторінці.
5. П'ятий крок Майстра: задайте ім'я для звіту. Автоматично це ім'я стане заголовком звіту.
6. Якщо при перегляді звіту знайдено деякі недоліки, відредагуйте звіт у режимі Конструктора.
7. Збережіть звіт.

Друк звіту здійснюється за допомогою команди *Печать* на стрічці *Предварительный просмотр*.

У режимі Конструктора макет стрічкового звіту має декілька розділів: заголовок, область даних, примітка тощо (рис. 11).

Заголовок отчета друкується на першої сторінці, якщо їх декілька, ***Примечание отчета*** – на останній, ***Колонтитулы*** верхній та нижній – на кожної сторінці. Нижній колонтитул містить вирази для вставки поточної дати та номера сторінки.

Область даних звіту містить заголовки стовпців (зазвичай це імена полів), та поля з даними. Якщо є групування записів, з'являються ***Заголовок группы*** и ***Примечание группы,*** де розміщено зноски з підсумками. Виконання розрахунків для підведення підсумків здійснюється за допомогою формул-виразів.

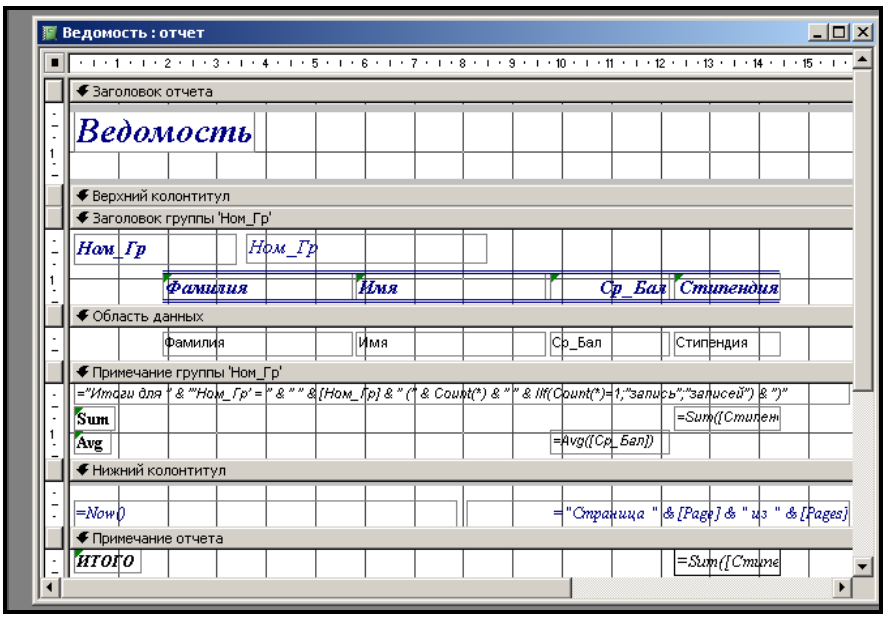


Рис. 11. Макет звіту *«Ведомость»* у Конструкторі

> ✍ Якщо формула знаходиться у розділі примітка групи, у розрахунках задіяні записі окремої групи. Якщо формула знаходиться у розділі примітка звіту у розрахунках задіяні всі записі.

Робота з макетом звіту у режимі Коструктора аналогічна роботі з макетом форми в Конструкторі. При цьому використовують команді на стрічці *ИНСТРУМЕНТЫ КОНСТРУКТОРА ОТЧЕТОВ / КОНСТРУКТОР, УПОРЯДОЧИТЬ, ФОРМАТ, ПАРАМЕТРЫ СТРАНИЦЫ* а також команди контекстного меню об'єктів. Приклад стрічкового звіту з підведеними підсумками представлено на рис. 12.

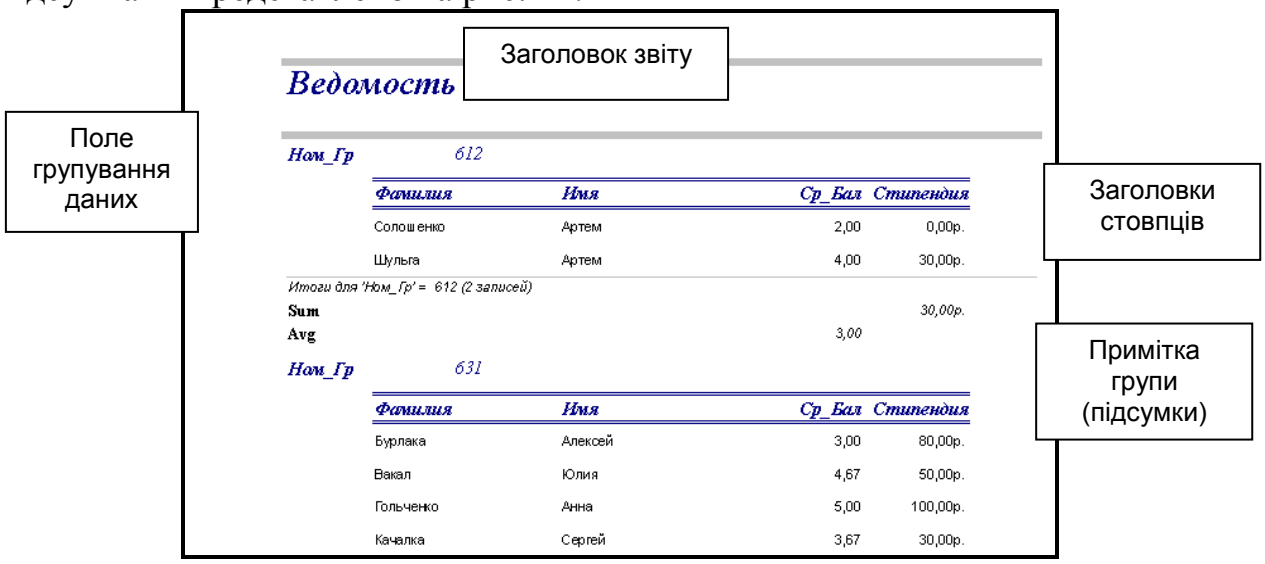


Рис. 12. Структура стрічкового звіту *«Ведомость»*

Створення листів для розсилки або поштових наклейок за допомогою Майстра.

1. Вибрати джерело даних для майстра – таблицю або запит.
2. Перший крок Майстра: встановити розмір наклейки.
3. Другий крок Майстра: вибір шрифтового оформлення.

4. Третій крок Майстра: створення тексту наклейки, вставка полів у текст.
5. Четвертий крок Майстра: вказати поле сортування
6. П’ятий крок Майстра: вказати ім’я звіту.
7. Відредагувати макет наклейки у режимі конструктора.

Лабораторна робота №6. Створення звітів

Мета: Придбання практичних навичок створення звітів різних типів.

Завдання.

Запустіть Access, відкрийте створену раніше БД «Деканат». Виконайте наступні вправи. Кожному створеному звіту дайте ім'я за змістом.

Створити за допомогою Майстра та відредагувати звіти:

1. Створити звіт «Успішність на потоці», що містить дані про екзаменаційну сесію по групах (номер групи, прізвище, ім'я, оцінки по трьох іспитах, заліки).

☞*Вказівки.* Звіт складати на основі двох таблиць «Студенти» та «Оцінки», згрупувати по полю «Номер групи», відсортувати за прізвищем.

2. Створити підсумковий звіт «Відомість про нарахування стипендії», що містить дані та підсумки про середній бал у сесію і нараховану стипендію по групах (номер групи, прізвище, ім'я, середній бал, стипендія).

☞*Вказівки.* Звіт складати на основі таблиці «Студенти» і запиту «Середній бал», звіт містить підсумкові дані про загальну суму стипендії і середній бал по групах.

3. Створити поштові наклейки для розсилання документації, указати прізвище, ім'я, повну адресу студента.
4. Створити лист-запрошення для зборів студентів які проживають у Сумах на додаткові заняття з правил дорожнього руху, додайте ілюстрацію.

? *Контрольні запитання.*

1. Призначення звітів.
2. Типи звітів у Access.
3. Способи створення звітів.
4. Алгоритм створення звіту за допомогою Майстра звітів.
5. Редагування звіту за допомогою Конструктора.
6. Редагування звіту у режимі Макет.
7. Групування даних у звіті.
8. Сортування даних у звіті.
9. Обчислення підсумків у звіті.
10. Додавання ілюстрацій у звіт.

2.7. Організація інтерфейсу для роботи з базою даних Access. Макроси

У БД Access зберігаються не лише дані, організовані в декількох взаємозв'язаних двовимірних таблицях, але й елементи інтерфейсу, запити, звіти і програмний код. Завдяки цій унікальній технології можна створювати додатки для роботи з БД, які придатні для будь-якого користувача та задовольняють всі його потреби. Макроси – це основний інструмент, який допомагає автоматизувати дії.

Макроси бувають окремими об’єктами і відображаються в області навігації в групі *Макросы*, їх називають ізольованими макросами. Ізольовані макроси можна використати будь де, багато разів, тобто вони є універсальними інструментами організації інтерфейсу.

Макроси також можуть бути пов'язані з об'єктами інтерфейсу, такими як кнопки, текстові поля, форми і звіти, їх називають макросами користувальницького інтерфейсу, або впровадженими макросами. За допомогою впроваджених макросів можна автоматизувати послідовності дій, таких як відкриття іншого об'єкта, застосування фільтра, запуск операції експорту і багато інших завдань. Впроваджені макроси стають частиною об'єкта або елементу управління, тому впроваджені макроси рекомендується створювати для автоматизації завдань, які специфічні для певної форми або звіту. Впроваджені макроси не відображаються в області навігації.

Кожен макрос складається з однієї або декількох макрокоманд. Залежно від поточного контексту деякі макрокоманди можуть бути недоступні.

До складу Access 2010-2013 входить новий конструктор для створення макросів. Умовні вирази дозволяють створювати більш складну логіку з використанням вкладених блоків «Якщо», «Інакше» і «Інакше якщо». Додавання конструкцій «Якщо», «Інакше якщо» і «Інакше» надала макросам ще більше гнучкості і можливості управління програмним потоком. Для запуску макрокоманд при виконанні визначених умов призначений блок «Якщо». Він прийшов на зміну стовпцю «Умова», який використовувався в більш ранніх версіях Access. У поєднанні з блоком «Якщо» також можна використовувати блоки «Інакше якщо» і «Інакше», аналогічно тому, як це робиться в інших мовах програмування. Команди, які можна додавати в блоки «Якщо», «Інакше якщо» і «Інакше», доступні в контекстному меню макрокоманди. Максимальний рівень вкладеності блоку «Якщо» - 10.

Створення Макросу.

1. Вибрати команду *СОЗДАНИЕ/Макрос*
2. Вказати послідовність макрокоманд та аргументи для кожної макрокоманди шляхом вибору варіантів зі списків, які розкриваються.
3. Перевірити роботу макросу *Выполнить!* Перед цім зберігши його.

Приклад створення макросу для додавання нового запису у форму представлений на рис. 13.

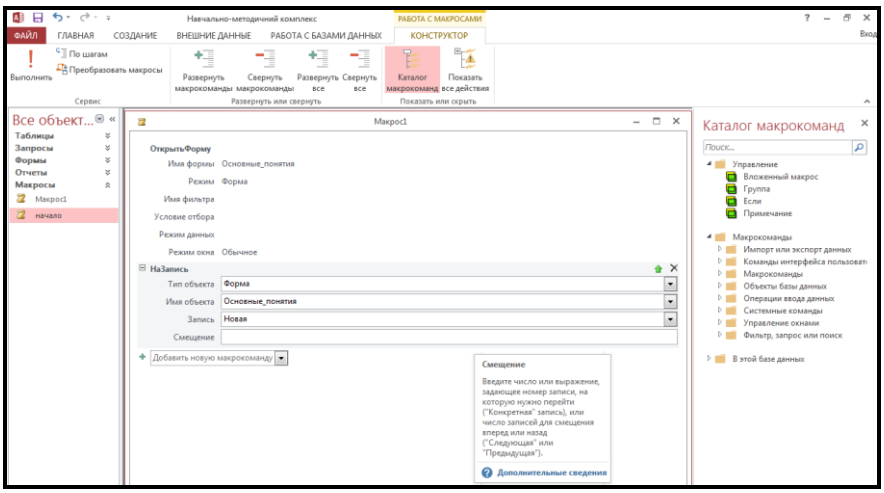


Рис. 13. Макрос у режимі Конструктора

Головна кнопкова форма використовується як головне меню додатка для роботи з БД та являє собою форму з кнопками, за допомогою яких можливо запускати макроси відкриття інших форм, звітів, друкувати звіти та інше. Завдяки головної формі окремі об'єкти БД зв'язані у єдину програму, роботу з ними автоматизовано та спрощено.

Головна кнопкова форма запускається насамперед при відкритті файлу БД, дає доступ до необхідних об'єктів та ховає від оператора об'єкти и панелі інструментів Access. Таким чином правильно оформлена програма-додаток для роботи з БД для конечного користувача має вигляд, який відрізняється від зовнішнього вигляду вікна БД в Access.

Щоб головна кнопкова форма була зрозумілою та естетично привабливою, треба використовувати небагато кнопок–елементів керування та приділити увагу оформленню. Пояснюючи надписи та малюнки, лінії що розділяють форму на логічні зони, а також кнопки-підменю, які розкривають наступні кнопкові форми-меню дозволяють ефективно та без помилок працювати з інформацією у БД.

Створення головної кнопкової форми можливе різними способами: по-перше у режимі Конструктора (як говорять «з нуля»), по-друге за допомогою диспетчера кнопкових форм, який запускається командою *СОЗДАНИЕ/Формы/Навигация.* Правила роботи в режимі Конструктора, користування панеллю елементів управління описано у розділі 2.5. «Форми». Основний елемент управління – кнопка, при натисненні кнопки запускається макрос для виконання той чи іншій операції, наприклад відкриття форми або звіту. Створити макрос можна самостійно, але цю операцію автоматизовано, у режимі Конструктора форм підключається необхідній Майстер.

Створення кнопок за допомогою Майстра:

1. Відкрити панель елементів управління форми. Зверніть увагу на те, підключений чи ні Майстер для створення елементів (кнопка *Использовать Мастера* активна).
2. Вибрати елемент *Кнопка,* розмістити кнопку в області даних шляхом перетаскування, має підключитися Майстер.

3. Перший крок Майстра: вибрати операцію, яка буде виконуватись при нажатій кнопці. Вибрати спочатку Категорію (*Переходы по записям, Обработка записей, Работа с формой, Работа с отчетом, Приложение, Разное)*. Після цього вибрати конкретну дію, наприклад, при роботі з формою – відкрити форму.
4. Другий крок Майстра: залежить від першого і конкретизує операцію, наприклад, вказати ім'я об'єкту.
5. Третій крок Майстра: вказати один з варіантів – виконати операцію для всіх записів або лише для відібраних записів
6. Четвертий крок Майстра: розмістити на кнопці текст або малюнок, який пояснює призначення кнопки.
7. П'ятий крок Майстра: вказати ім'я кнопки, за замовчанням *Кнопка0, Кнопка1…*
8. Налагодити роботу кнопки, змінити деякі параметри можна через контекстне меню кнопки: команда *Свойства/ Макет, Данные, События, Другие*.

✍ Перевірити роботу кнопки потрібно у режимі форми.

Створення елементу керування з впровадженим макросом

Щоб створити впроваджений макрос для події *Нажатие кнопки* необхідно відкрити форму або звіт в режимі конструктора, викликати контекстне меню створеної командної кнопки, задати *Обработка событий … / Макросы / Добавить новую макрокоманду …*

Якщо визвати контекстне меню області даних, й додати макрос аналогічно, він буде спрацьовувати при клацанні просто на формі.

Також зручно користуватись у режимі Конструктора форм сторінкою властивостей конкретного елемента. Наприклад, *Страница свойств / Форма / События / Открытие / Макросы / …* Такий макрос буде спрацьовувати при відкриванні форми.

Дуже часто виникає необхідність створити елемент керування у формі або звіти, який спрацьовує при виконанні умови. Для цього використовують ***впроваджені макроси з умовою***. Наприклад, впроваджені макроси для створення кнопок фільтру при організації форми у вигляді каталогу або словника (рис. 14). Ім'я поля, таблиці, які вказано в умові, має обов'язково бути серед списку полів джерела даних.

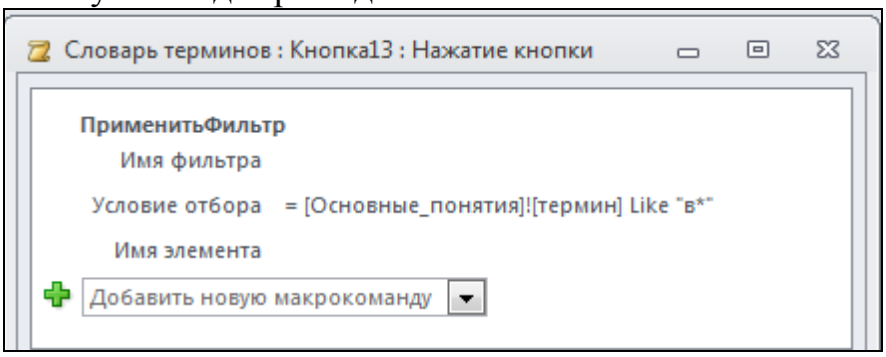


Рис. 14. Макрос впроваджений в форму «Словарь терминов» для застосування фільтру за абеткою на літеру В

Впроваджений макрос з умовою використовують для перевірки логіну й паролю при розділеному доступі к БД, для спливання Вікна повідомлень, а також для інших команд інтерфейсу користувача, які доступні з каталогу макрокоманд (рис. 15).

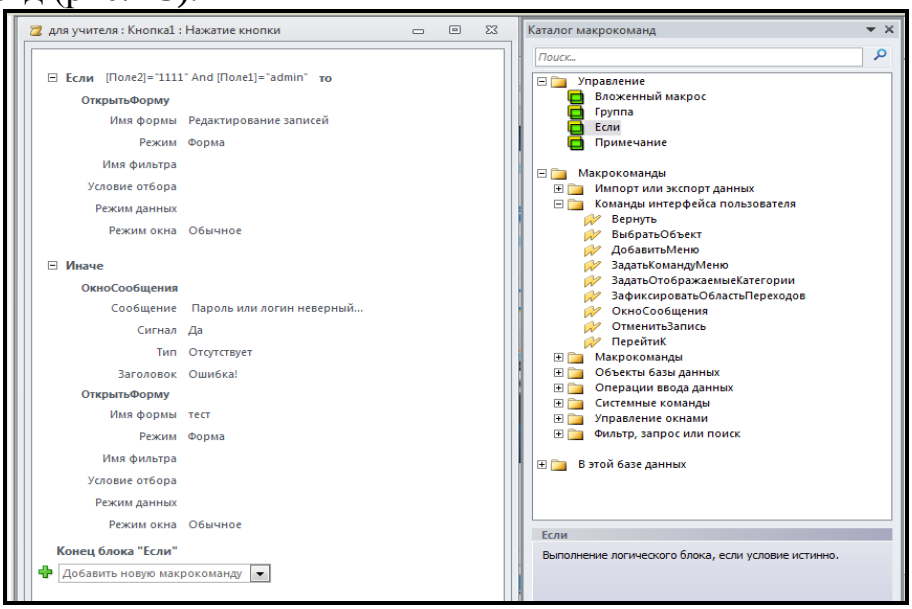


Рис. 15. Макрос для кнопки «Войти»

Автоматичній запуск головної кнопкової форми при старті

Для того щоб головна кнопкова форма запускалась першою при відкритті файлу БД, треба змінити параметри запуску: Файл / *Параметры / Текущая база даних.* У діалоговому вікні для параметру *Форма просмотра* вказати ім'я головної кнопкової форми, та якщо необхідно інші параметри. Для скасування параметрів запуску, наприклад при неможливості операцій та відсутності доступу до вікна БД, треба тримати на клавіатурі клавішу SHIFT при запуску.

Лабораторна робота №7. Створення кнопкової форми та макросів

Мета: Придбання практичних навичок оформлення додатка Access, закріплення навичок створення форм.

Завдання.

Запустіть Access, відкрийте створену раніше БД «Деканат». Виконайте наступні вправи.

1. Створити головну кнопкову форму, включивши наступні елементи:
   - заголовок «Навчальна база даних «Деканат»;
   - малюнок, що відповідає змісту бази даних;
   - примітку: автор програми – Прізвище студента
2. Створити кнопки для автоматизації роботи з БД ***за допомогою Майстра***:
   2.1. кнопку для відкриття форми «Студенти й оцінки»;
   2.2. кнопку для відкриття звіту «Відомість про нарахування стипендії»;
   2.3. кнопку виходу з програми;
   2.4. кнопку «Пошук» для пошуку запису на прізвище;

☞ *Вказівки.* Спочатку створити запит з параметром для пошуку даних за Прізвищем, потім для нього зручну форму, потім кнопку для відкриття форми.

3. Створити макроси для роботи з БД за допомогою Конструктора
   - 3.1. створити ізольований макрос для додавання запису у форму «Студенти»;
   - 3.2. у головній кнопковій формі створити кнопку «Додати запис студента» для запуску цього макросу;
   - 3.3. створити кнопки з впровадженими макросами для форми «Студенти й оцінки» для застосування фільтру – прізвище починається на літеру Л, К;
   - 3.4. створити впроваджений макрос для області даних для зняття фільтру.
4. Забезпечити автоматичний запуск головної кнопкової форми при старті.
5. Задати пароль для відкриття БД.

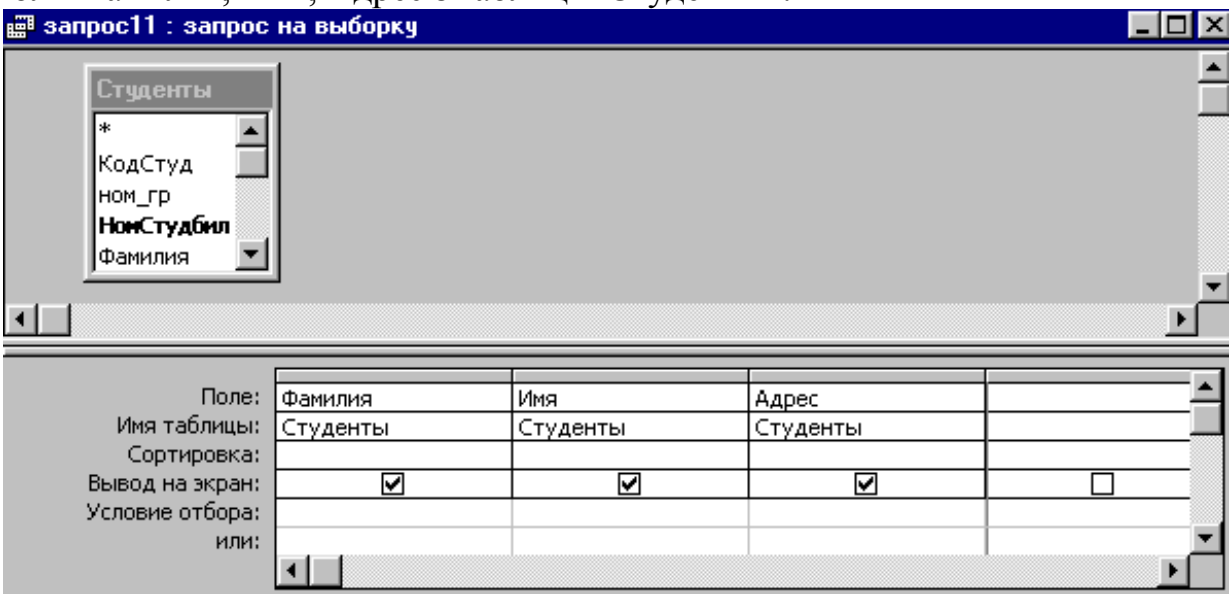
? *Контрольні запитання.*

1. Призначення головної кнопкової форми.
2. Особливості головної кнопкової форми.
3. Створення головної кнопкової форми.
4. Макроси. Види макросів.
5. Створення макросів за допомогою Майстра.
6. Створення макросів за допомогою Конструктора.
7. Створення впроваджених макросів.
8. Способи завдання умови у макросах
9. Редагування макросів.
10. Автоматичний запуск форми при старті.

Додаток А.
Приклади запитів SQL й QBE

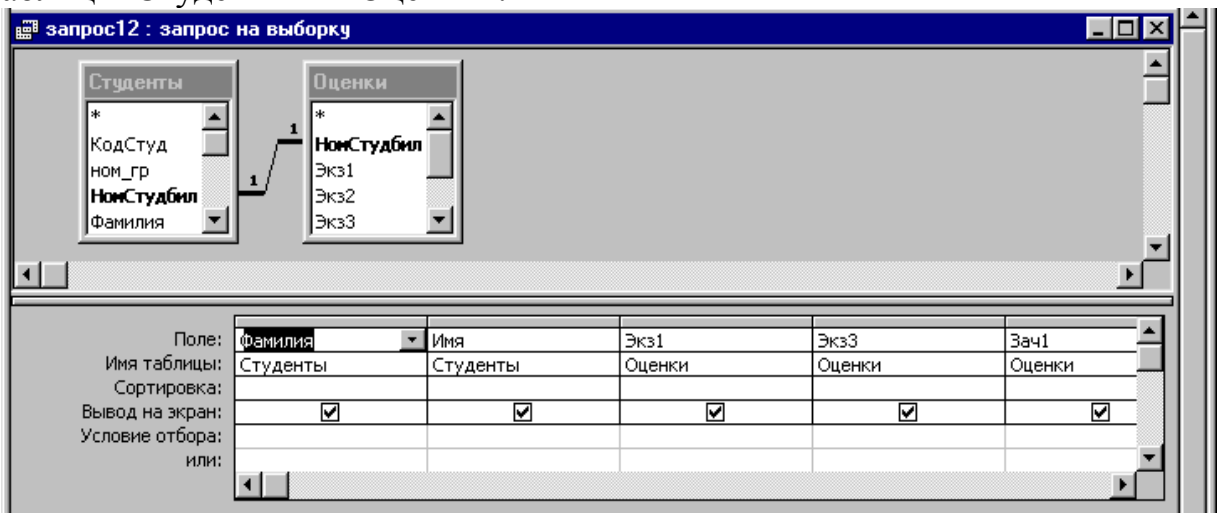
Розв’язки завдань з лабораторної роботи № 3

Запит 11 виводить на екран список всіх студентів з БД, використовуючи поля Фамилия, Имя, Адрес з таблиці «Студенты».



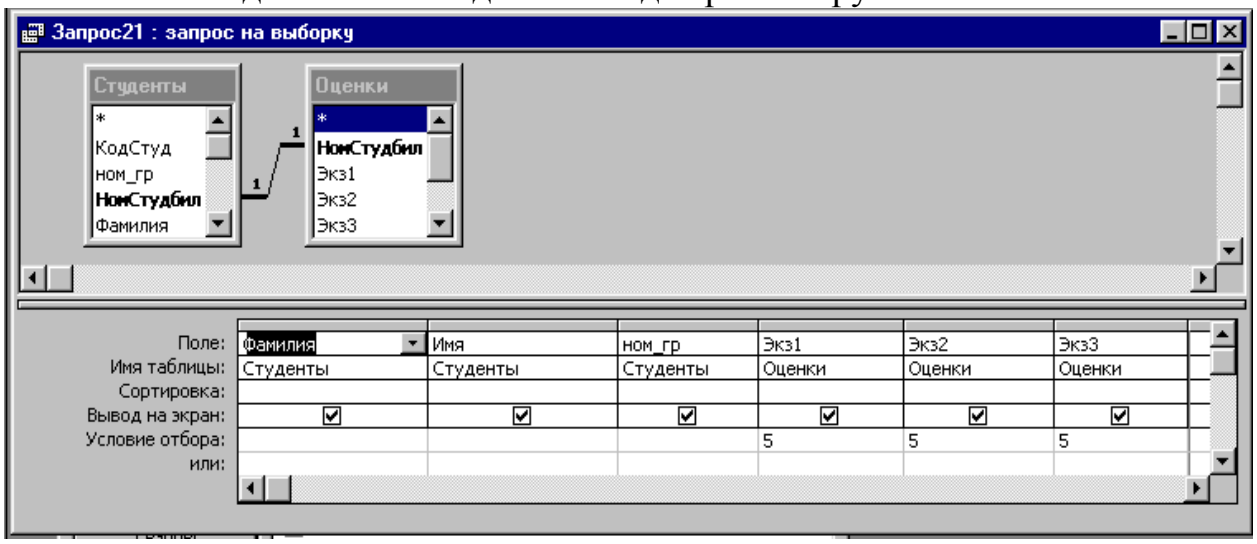
SELECT Студенты.Фамилия, Студенты.Имя, Студенты.Адрес
FROM Студенты;

Запит 12 виводить прізвища, імена і оцінки студентів використовуючи дві таблиці «Студенты» і «Оценки».



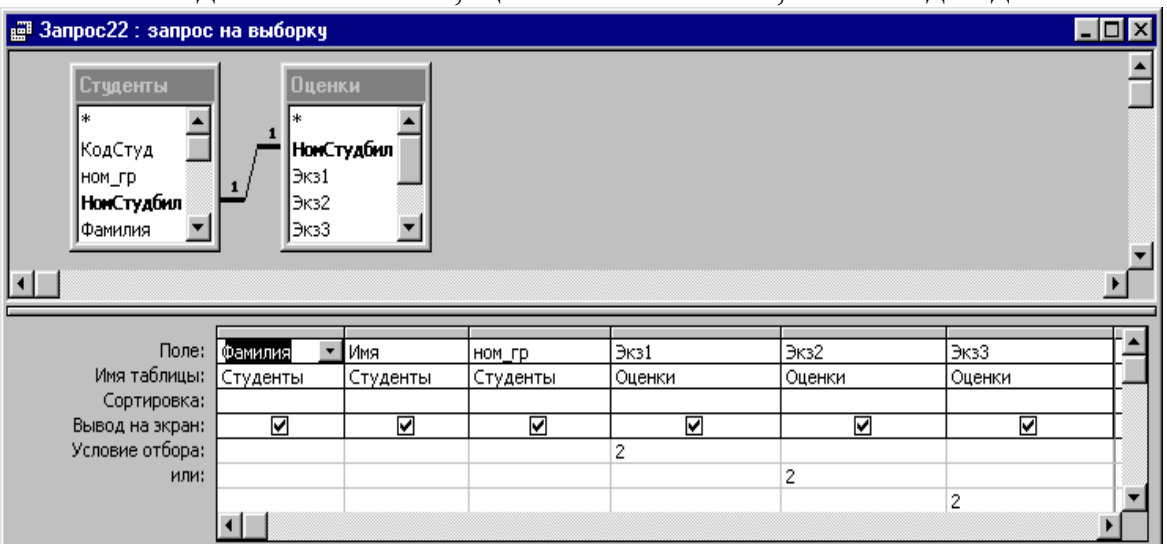
SELECT Студенты.Фамилия, Студенты.Имя, Оценки.Экз1, Оценки.Экз3, Оценки.Зач1
FROM Студенты INNER JOIN Оценки ON Студенты.НомСтудбил = Оценки.НомСтудбил;

Запит 21 виводить списки відмінників для різних груп.



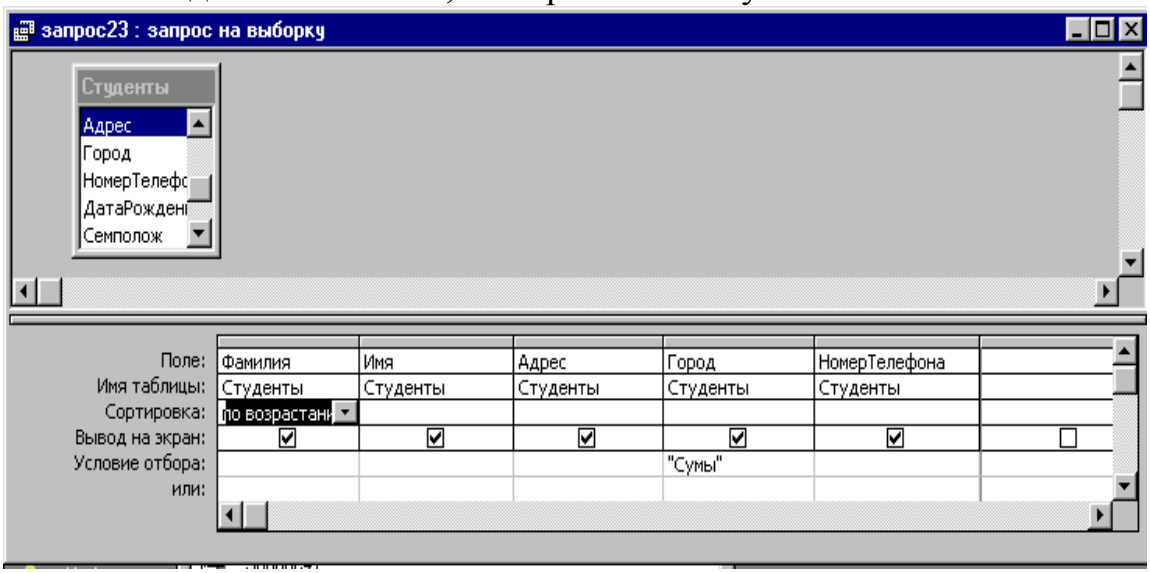
SELECT Студенты.Фамилия, Студенты.Имя, Студенты.ном_гр, Оценки.Экз1, Оценки.Экз2, Оценки.Экз3
FROM Студенты INNER JOIN Оценки ON Студенты.НомСтудбил = Оценки.НомСтудбил
WHERE (((Оценки.Экз1)=5) AND ((Оценки.Экз2)=5) AND ((Оценки.Экз3)=5));

Запит 22 виводить списки тих, що не склали іспит, хоч би одна двійка.



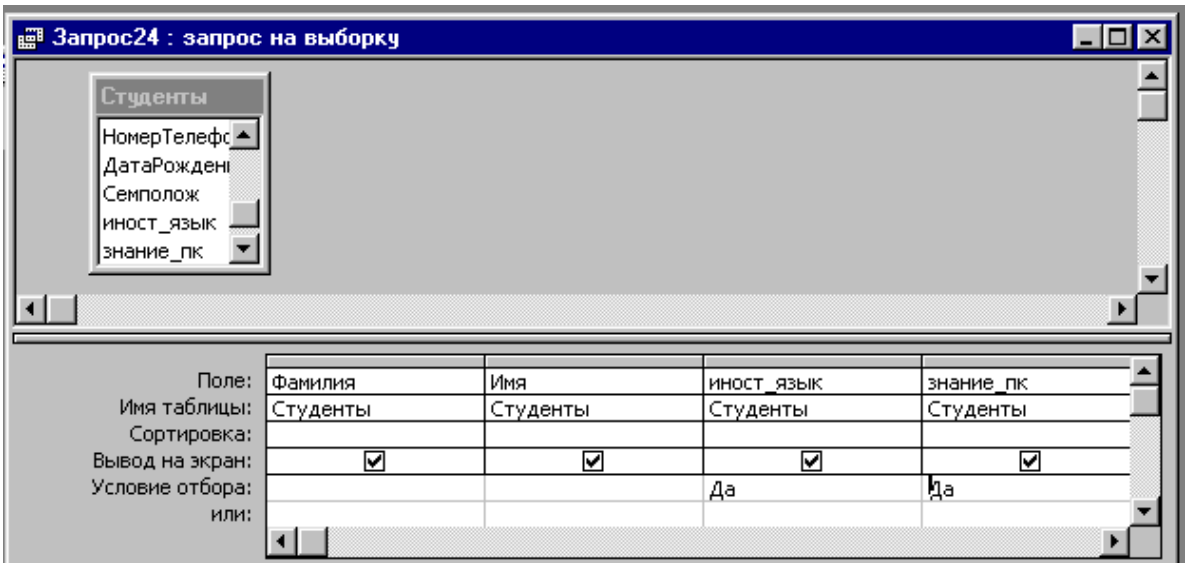
SELECT Студенты.Фамилия, Студенты.Имя, Студенты.ном_гр, Оценки.Экз1, Оценки.Экз2, Оценки.Экз3
FROM Студенты INNER JOIN Оценки ON Студенты.НомСтудбил = Оценки.НомСтудбил
WHERE (((Оценки.Экз1)=2)) OR (((Оценки.Экз2)=2)) OR (((Оценки.Экз3)=2));

Запит 23 виводить список тих, хто проживає в Сумах.



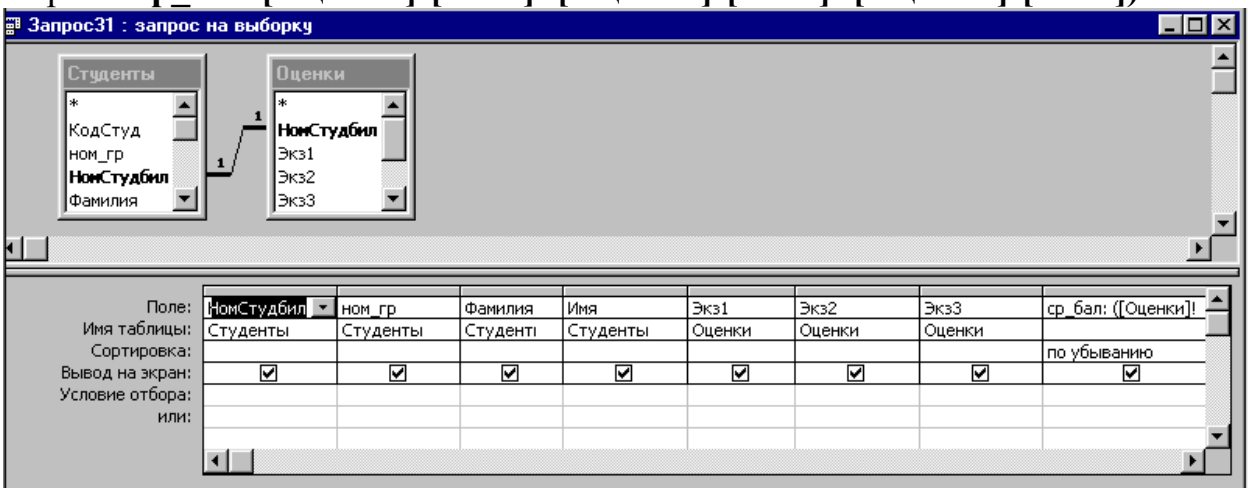
SELECT Студенты.Фамилия, Студенты.Имя, Студенты.Адреса, Студенты.Город, Студенты.НомерТелефона
FROM Студенты
WHERE (((Студенты.Город)="Сумы"))
ORDER BY Студенты.Фамилия;

Запит 24 виводить список студентів, що володіють комп'ютерною технікою та іноземною мовою.



SELECT Студенты.Фамилия, Студенты.Имя, Студенты.иност_язык, Студенты.знание_пк
FROM Студенты
WHERE (((Студенты.иност_язык)=Yes) AND ((Студенты.знание_пк)=Yes));

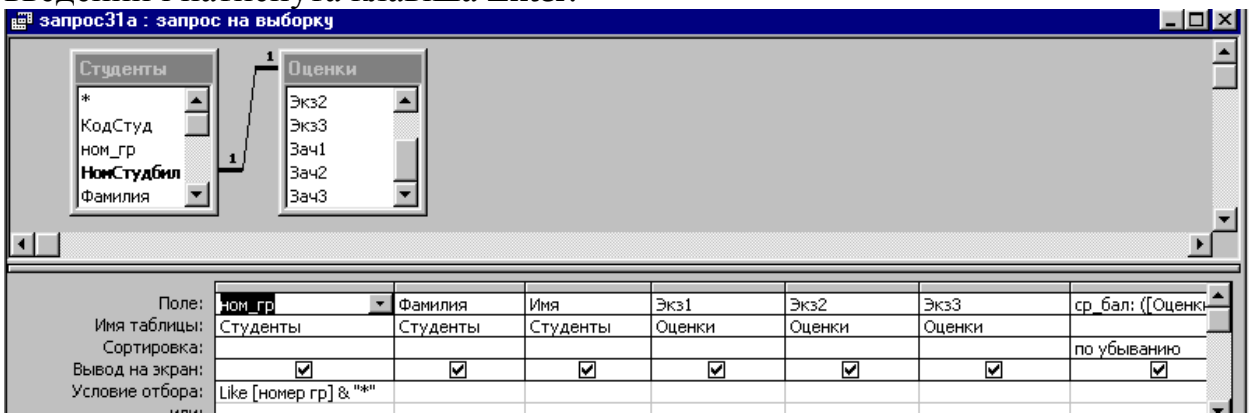
Запит 31 виводить дані про успішність студентів (середній бал на іспиті) вираз для розрахунку середнього балу будуємо за допомогою Майстра Виразів **ср_бал:[Оценки]![Экз1]+[Оценки]![Экз2]+[Оценки]![Экз3])/3**



SELECT Студенты.НомСтудбил, Студенты.ном_гр, Студенты.Фамилия, Студенты.Имя, Оценки.Экз1, Оценки.Экз2, Оценки.Экз3, ([Оценки]![Экз1]+[Оценки]![Экз2]+[Оценки]![Экз3])/3 AS ср_бал
FROM Студенты INNER JOIN Оценки ON Студенты.НомСтудбил = Оценки.НомСтудбил
ORDER BY ([Оценки]![Экз1]+[Оценки]![Экз2]+[Оценки]![Экз3]) /3 DESC;

Запит 31а На базі попереднього запиту будуємо запит з параметром, як параметр запрошується номер групи*:*

Вираз ***Like [номер гр] & "*"*** дозволяє вивести всі дані якщо параметр не введений і натиснута клавіша Enter.



SELECT Студенты.ном_гр, Студенты.Фамилия, Студенты.Имя, Оценки.Экз1, Оценки.Экз2, Оценки.Экз3, ([Оценки]![Экз1]+[Оценки]![Экз2]+[Оценки]![Экз3])/3 AS ср_бал
FROM Студенты INNER JOIN Оценки ON Студенты.НомСтудбил = Оценки.НомСтудбил
WHERE (((Студенты.ном_гр) Like [номер гр] & "*"))
ORDER BY ([Оценки]![Экз1]+[Оценки]![Экз2]+[Оценки]![Экз3]) /3 DESC;

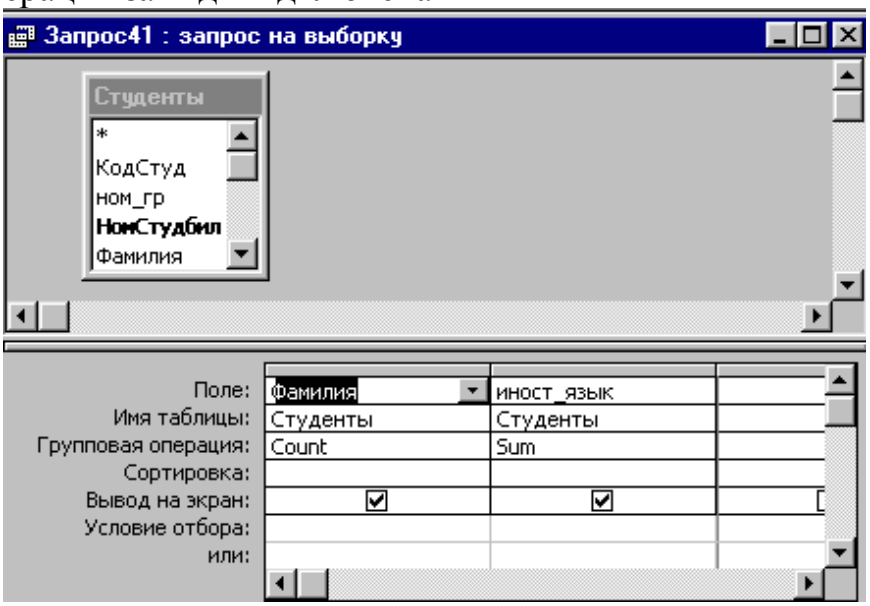
Запит 32 виводить список кращих 5 студентів по успішності.

Створюється аналогічно попередньому, при цьому змінюються властивості всього запиту в режимі Конструктора через контекстне меню верхнього поля запиту, набір значень обмежується. У режимі SQL додається інструкція TOP 5.

Розв’язки завдань з лабораторної роботи № 4

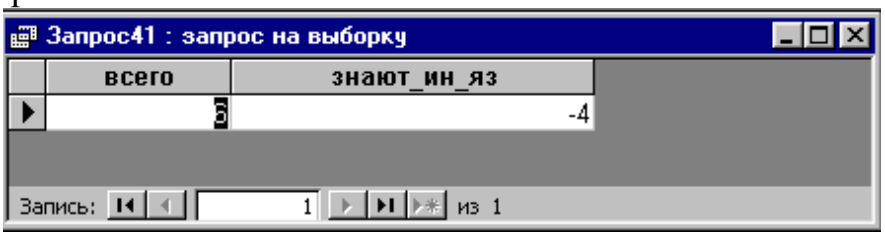
Запит 41 для підрахунку кількості студентів, які володіють іноземною мовою.

У режимі Конструктора підсумкові запити містять мало полів, кнопка «Групові операції» завжди підключена



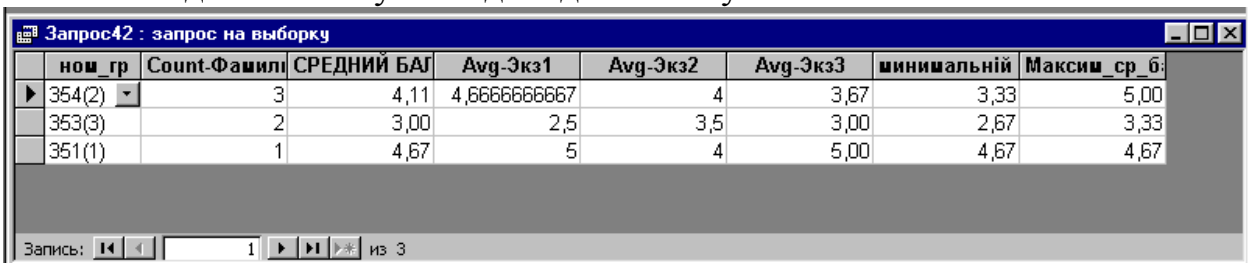

```
SELECT Count(Студенты.Фамилия) AS [Count-Фамилия],
Sum(Студенты.иност_язык) AS [Sum-иност_язык]
FROM Студенты;
```

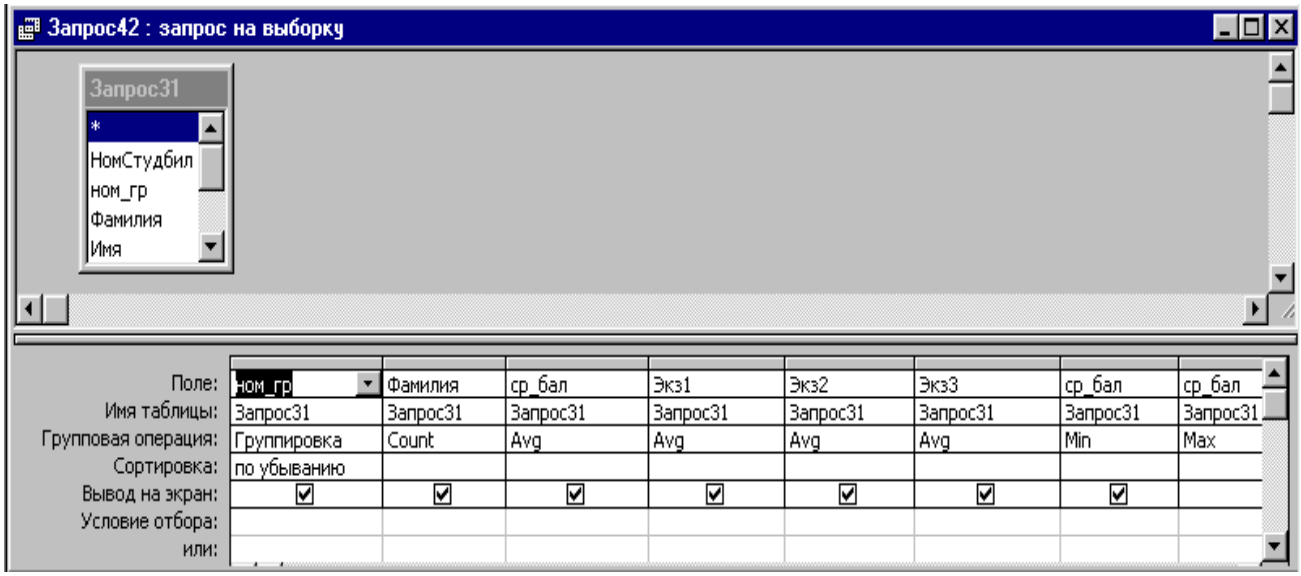
Підсумовування по логічному полю завжди дає суму значень із знаком мінус, оскільки фактично значення «Істина» зберігається в БД як: -1, а значення «Брехня» як 0.



Запит 42 виводить загальну кількість студентів, успішність в цілому, успішність по окремих предметах, мінімальне значення успішності, максимальне значення успішності по окремих групах.

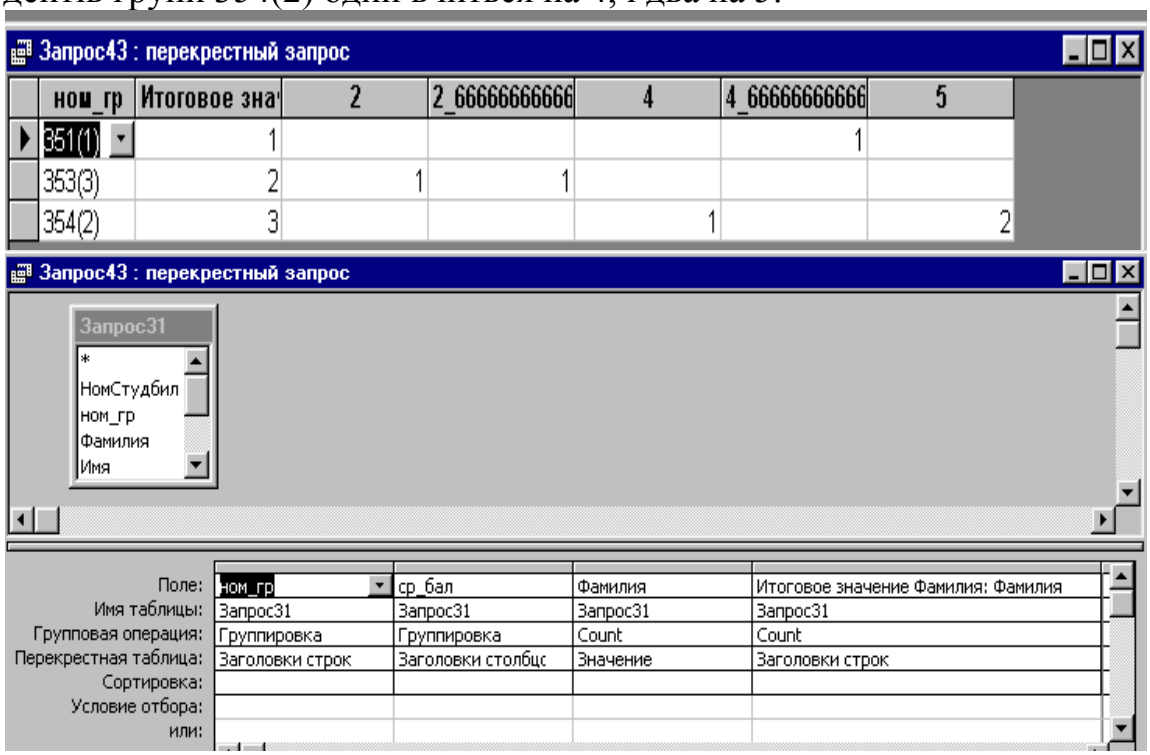
Запит про успішність заснований на запиті 31, це приклад вкладеного запиту, при виконанні Запиту 42 спочатку виконується запит31, таким чином, маємо завжди свіжі актуальні дані для аналізу.





SELECT Запрос31.ном_гр, Count(Запрос31.Фамилия) AS [Count-Фамилия], Avg(Запрос31.ср_бал) AS [Avg-ср_бал], Avg(Запрос31.Экз1) AS [Avg-Экз1], Avg(Запрос31.Экз2) AS [Avg-Экз2], Avg(Запрос31.Экз3) AS [Avg-Экз3], Min(Запрос31.ср_бал) AS [Min-ср_бал], Max(Запрос31.ср_бал) AS [Max-ср_бал]
FROM Запрос31
GROUP BY Запрос31.ном_гр
ORDER BY Запрос31.ном_гр DESC;

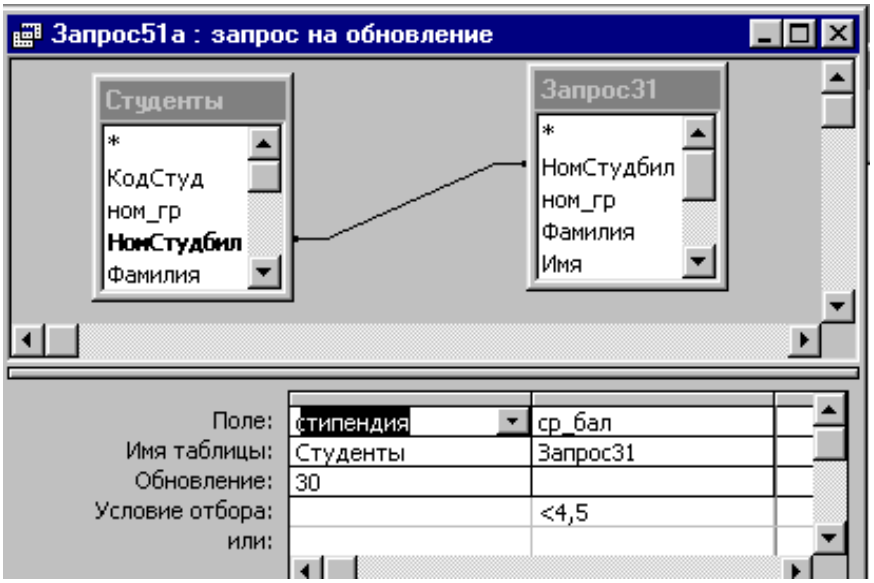
Запит 43 виводить дані про успішність окремих груп, це перехресний запит. Перехресний запит зручніше створювати за допомогою Майстра перехресних запитів, в основі цього запиту лежить запит 31. У режимі перегляду таблиця відображає згруповані дані про успішність в групах. Наприклад з трьох студентів групи 354(2) один вчиться на 4, і два на 5.



TRANSFORM Count(Запрос31.Фамилия) AS [Count-Фамилия]
SELECT Запрос31.ном_гр, Count(Запрос31.Фамилия) AS [Итоговое значение Фамилия]
FROM Запрос31
GROUP BY Запрос31.ном_гр
PIVOT Запрос31.ср_бал;

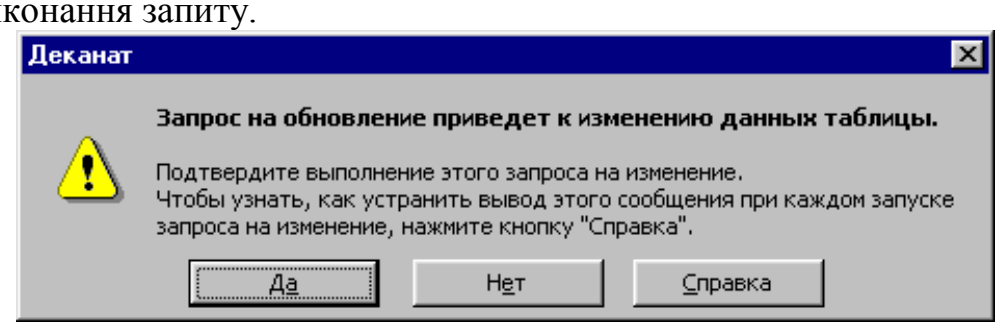
Запит 51 - запит на призначення стипендії в 30 грн. тим у кого середній бал нижче 4,5.

Заснований на таблиці Студенты і запиті 31 про успішність, зверніть увагу – таблиця і запит повинні бути зв'язані, для цього включите в запит 31 ключове поле НомСтудбил



UPDATE Студенты INNER JOIN Запрос31 ON Студенты.НомСтудбил = Запрос31.НомСтудбил SET Студенты.стипендия = 30
WHERE (((Запрос31.ср_бал)<4.5));

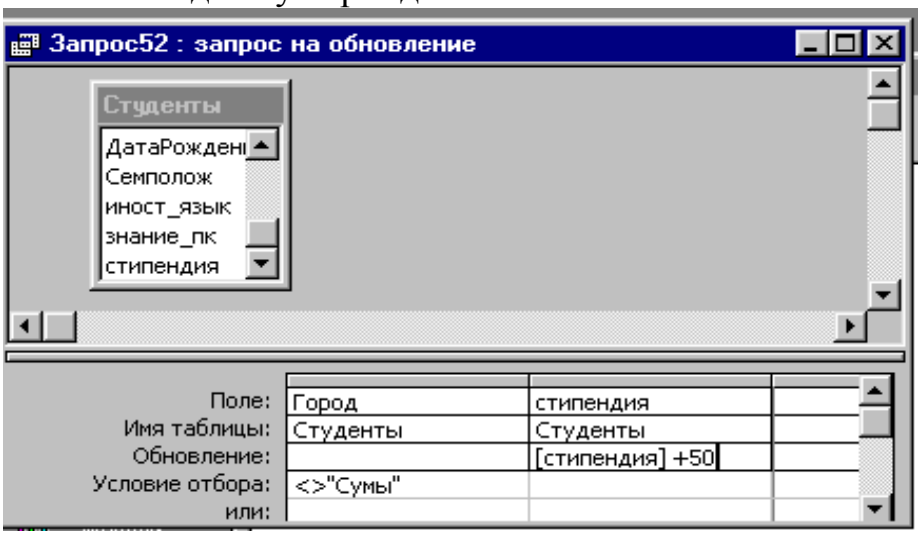
Виконання запиту.



При підтвердженні операції обновляються базові таблиці. Такі запити звичайно не зберігають, оскільки їх потрібно виконувати один раз.

Запит 52 для підвищення стипендії студентам, які не проживають в Сумах.

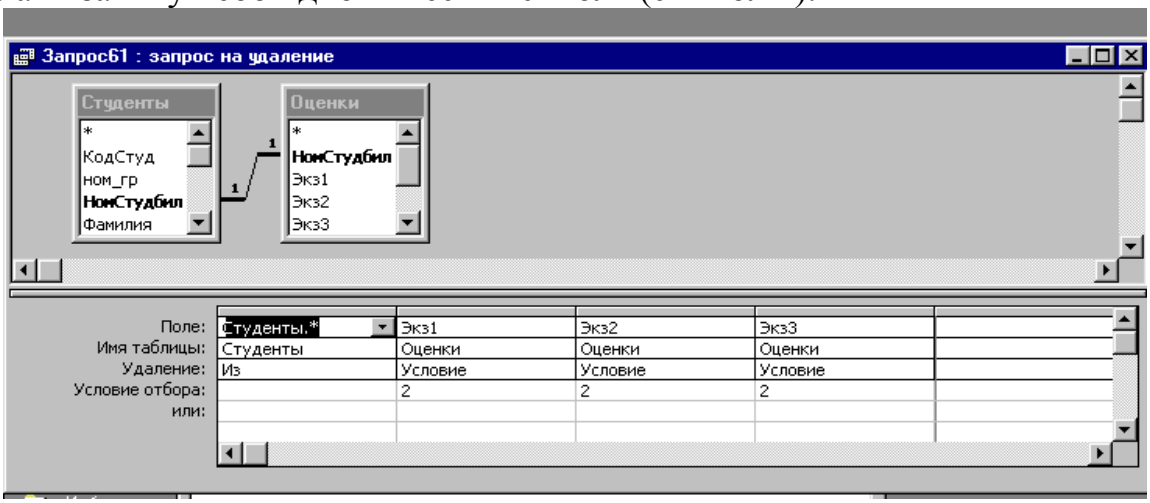
При оновленні значень полів з урахуванням попередніх значень в полі, ім’я поля повинне входити у вираз для оновлення.



UPDATE Студенты SET Студенты.стипендия = [стипендия]+50
WHERE (((Студенты.Город)<>"Сумы"));

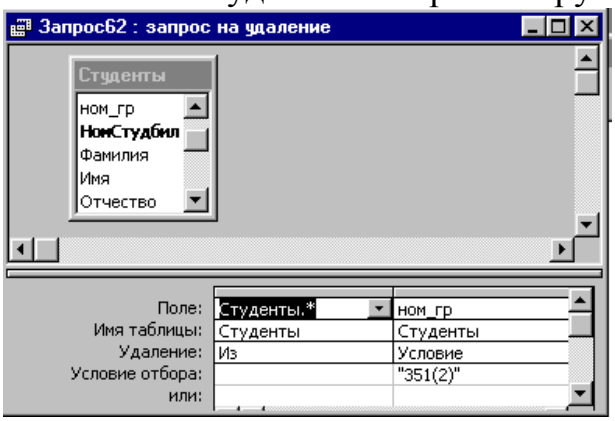
Запит 61 на видалення студентів круглих двієчників.

Для спрацьовування запитів на видалення необхідно підключити каскадне видалення на схемі даних у властивостях зв'язку, зверніть увагу, що в бланк запиту необхідно винести всі поля (символ *).



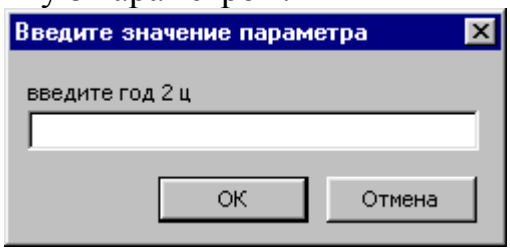
DELETE Студенты.*, Оценки.Экз1, Оценки.Экз2, Оценки.Экз3
FROM Студенты INNER JOIN Оценки ON Студенты.НомСтудбил = Оценки.НомСтудбил
WHERE (((Оценки.Экз1)=2) AND ((Оценки.Экз2)=2) AND ((Оценки.Экз3)=2));

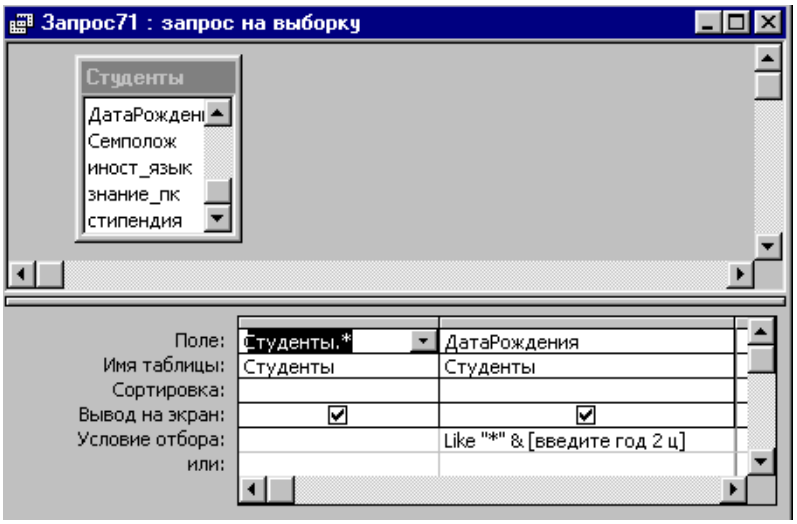
Запит 62 на видалення записів студентів конкретної групи.



DELETE Студенты.*, Студенты.ном_гр
FROM Студенты
WHERE (((Студенты.ном_гр)="351(2)"));

Запит 71 з параметром для виведення списку студентів певного року народження. При виконанні запрошує параметр, в даному випадку рік народження, в [ ] вказівка для введення параметру.
Початок виконання запиту з параметром.





SELECT Студенты.*, Студенты.ДатаРождения
FROM Студенты
WHERE (((Студенты.ДатаРождения) Like "*" & [введите год 2 ц]));

Додаток Б.

Основні інструкції мови SQL

	вказівка	опис
основні інструкції	*SELECT поле1, поле2,…полеN FROM табл1*	вибір *N* полів з вказаної таблиці
	WHERE умова	задає умову
	IN повне ім'я файлу бази даних	для відбору даних з неактивної бази даних
	SELECT табл1.поле1, табл2.поле1.. .табл2.полеN FROM табл1 INNER JOIN табл2 ON табл1.поле1=табл2.поле1	вибір полів з двох таблиць зв'язаних по полю1
	DELETE FROM табл1	видалення записів
	UPDATE табл1 SET поле1=поле1, вираз	оновлення записів
	ORDER BY поле	сортування записів по полю за збільшенням. .
	ORDER BY поле DESC	сортування записів по полю за зменшенням
	INSERT INTO табл1 SELECT…	вставка в таблицю вибраних записів
предикати	ALL	відображати всі записи
	TOP N	відображати перші N-штук записів
	DISTICT	відображати унікальні значення
	DISTICTROW	відображати унікальні записи (для зв’язаних таблиць)

Предметний покажчик

Б

базова таблиця, 6, 24
БД, 5
бланк фільтра, 32

В

введення даних, 27
видалення записів, 27
вікно БД, 24
вкладений запит, 37
властивості запиту, 40

Д

динамічний набір даних, 36
діапазон значень, 9
додавання запису, 24

З

забезпечення цілісності даних, 28
запис, 7
запит, 6, 42, 44, 45
запит QBE, 37, 38
запит SQL, 37
запит на вибірку, 36
запити на зміну, 36, 44
звіт, 6, 52

І

ім'я поля, 11, 12, 33
індекс, 8, 12

К

каскадне видалення, 28
каскадне оновлення, 28
ключове поле, 13
комбінація клавіш, 31, 32, 59
Конструктор, 23, 26, 38

М

Майстер, 13, 17, 38, 42, 57
макрос, 7
маска, 12
міжтабличні зв'язкі, 27
модуль, 7

О

опис, 12

П

параметричні запити, 36, 39
переіндексация, 13
перехресний запит, 36, 43
підпис, 12
підсумкові запити, 36, 41
поле, 7, 47
поле зі списком, 13
пошук записів, 31
пошук і заміна, 32

Р

редагування запису, 24
розмір поля, 9, 10, 12

С

сортування, 31
статичний набір даних, 37
створення БД, 7
структура, 22, 23, 25, 26, 29, 45
СУБД, 5
схема даних, 27

Т

тип відношення, 8
тип об'єднання, 28
тип поля, 9, 10, 11, 12, 21, 42
точність, 8, 10, 12
транзакція, 22

У

умова, 43

Ф

фільтр, 32
форма, 6
формат поля, 12

Ш

ширина стовпців, 31

Література

1. Баженов В.А. Венгерский П.С. и др. Інформатика, підручник для студентів вищих закладів освіти, К: Каравела, 2008.– 640 с.
2. Пасько В. Access 97. Практическое руководство пользователя. К.. «Наука», 1998, – 497 с.
3. Праг Д., Ірвін М. MS Access 2000 Библия пользователя, М.: 2001.
4. Роджер Д. Использование Access97, спец. Видавництво К-М-С-Пет: 1998
5. Симонович С.В. и др. Информатика. Базовий курс, учебник для ВУЗов. СПб: «Питер», 2000. – 640 с.
6. Тимошок Т.В. MS Access 2002, Краткое руководство: М.:Изд.дом «Вильямс» «Диалектика», 2004. –272 с.
7. Шамшина Н.В. Практикум із MS Access: Навчальний посібник. – Суми: Вид-во СумДПУ ім. А.С. Макаренка, 2009. – 75 с.

Навчальне видання

Шамшина Наталія Володимирівна

ІНФОРМАТИКА.
СИСТЕМА УПРАВЛІННЯ БАЗАМИ ДАНИХ
MICROSOFT ACCESS

Навчальний посібник

Суми: СумДПУ, 2015 р.
Свідоцтво ДК № 231 від 02.11.2000 р.

Відповідальний за випуск
Комп'ютерний набір Н.В. Шамшина
Комп'ютерна верстка Н.В. Шамшина

Здано в набір 9.02.15. Підписано до друку 19.02.15
Формат 60х84/14. Гарн. Tims. Друк ризогр.
Папір друк. Умовн. друк. арк. 4,05. Обл.-вид.арк.3,7
Тираж 50 прим. Вид. № 97.

СумДПУ ім. А.С. Макаренка
40002, м. Суми, вул.. Роменська, 87
Виготовлено на обладнанні СумДПУ ім. А.С. Макаренка